O`ZBEKISTON RESPUBLIKASI OLIY VA O`RTA MAXSUS TA`LIM VAZIRLI

ANDIJON DAVLAT UNIVERSITETI

“Informatika” kafedrasi

“Dasturlash texnologiyasi” fanidan

M A ` R U Z A L A R
M A T N I

Andijon-2008

“Dasturlash texnologiyasi” fani bo`yicha ma`ruzalar matni “I va IAT” yo`nalishi talabalari uchun mo`ljallangan.

Ushbu ma`ruzalar matnida dastur tuzishning texnologiyasi bayon qilingan bo`lib, programmalashtirish yo`li, strukturali programmalash, programmalarni loyihalashtirish, amaliy programmalashtirish, programma samaradorligi, programmaning asosiy harakteristikalari kabi mavzular keltirilgan.

Ma`ruza matnlari “Informatika” kafedrasi uslubiy seminarida ko`rib chiqilgan (__ -son yig`ilishi bayoni «___» _____ 2008 yil).

Tuzuvchi: t.f.n. Mahkamov M.K

1 – Mavzu: Programmalashtirish yo`li. Izohlar, bo`sh satrlar qoldirish, bo`sh xonalar nomlash, tartibli nomerlash. O`zgaruvchilarning nomini tanlash, fayllar nomi, qisqartirishlar, operatorlarni joylash.

R E J A :

1. **Izohlar.**
2. **Bo`sh satrlar qoldirish. Bo`sh xonalar.**
3. **Fayllar nomi. Qisqartirishlar.**

**Tayanch so`zlar:** *Izohlar, bo`sh satrlar qoldirish, bo`sh xonalar nomlash, tartibli nomerlash. O`zgaruvchilarning nomini tanlash, fayllar nomi, qisqartirishlar, operatorlarni joylash.*

Izoh yoki sharh.

Tushuntirish izohlariga ega bo`lgan dasturlarni to`g`rilash yoki tuzatish osonroq, chunki ular dastur bilan ishlash uchun qo`shimcha ma`lumotlarni jamlaydi. Dastur tuzuvchi ko`pincha boshqa tuzuvchining dasturini ko`rib chiqayotgan vaqtda, xatolarni tuzatishda, yoki dastur mantiqni kuzatayotgan vaqtda juda ko`p vaqt sarflaydi. Bu holatda u o`zining me`yorlangan vaqtini juda katta qismni sarflab qo`yadi.

Izohlanmagan dastur - bu taxminimizcha dastur tuzuvchining eng katta xatosidir. Dastur yozish jarayonida izoh berish yaxshi qoidadir. Kelajakda buni hamma dasturchilar hisobga oladilar, degan umiddamiz. Izohning yozishning asosiy qonuni - bu ko`proq izoh berishdir. Albatta, har bir kishi dasturdagi izohni o`qiyotib, shu dastur haqida ko`proq tushunchaga ega bo`ladi. SHuning uchun qancha kerak bo`lsa, shuncha izoh bering.

Izoh berishdan maqsad - dasturni tushunishni osonlashtirishdir.

Ular dasturni kodlashtirganidek yaxshi o`ylagan va ishlab chiqilgan bo`lishi kerak. Izohlar loyihalash va sozlash jarayoni vaqtida va albatta keyinchalik ham kerak bo`ladi. Lekin dastur tugagandan keyin izoh berish foydasiz va ma`nosizdir. Izoh berishni 3 ta shakli bor:

kiritishda, mundarija va tushuntirishda.

Kiritish izohlari.

har bir dastur, kichik dastur va jarayonlar nima vazifani bajarishligi haqidagi izohlardan boshlanadi. Kiritish izohlaridagi ma`lumotlar quyidagilardan iborat bo`ladi:

1. Dasturdan maqsad.
2. Dasturni chaqirish va undan foydalanish haqidagi ko`rsatkichlar.
3. Asosiy massiv va o`zgartirgichlarning ro`yxati va vazifasi.
4. Kiritish - chiqarish haqidagi ko`rsatkichlar. Hamma fayllar ro`yxati.
5. Foydalanilayotgan barcha qism dasturlar ro`yxati.
6. Foydalanilgan matematik metodlar va ular haqida ma`lumotga ega bo`lgan adabiyotlar ro`yxati.
7. Dasturni bajarilgan vaqti haqida ma`lumot.
8. Kerak bo`lgan xotira hajmi.
9. Operatorga maxsus ko`rsatmalar.
10. Muallif haqida ma`lumot.
11. Dastur yozilgan sana.

Mundarija.

Agar dastur katta hajmga ega bo`lsa, uning boshiga izoh shaklida mundarija qo`yish maqsadga muvofiqdir. Mundarija o`z ichiga dastur modulining nomini, joyini va funktsiyasini jamlaydi.

Modullar ismiga, izoh yoki funktsiyalarga ko`rsatmalar bilan ta`minlangan bo`lishi kerak.

Tushuntirsh izohlari.

Dasturning izohlarsiz tushunib bO`lmaydigan qismlarga tushuntirish izohlari beriladi. Dasturning mantig`ini tushunishdan oldin sonlar yoki shartli operatorlardan

ilgari ular haqidagi ko`rsatmalardan iborat izohlar chiqib turishi kerak. YUqori pog`onada yozilgan 10 ta qatorga 1 ta izohlash qatori o`rtacha meoyor hisoblanadi. Izohlar operatorlar yaratayotgan harakatlarni taoriflamay, shu operator guruhlari operatorlarini aytib o`tish kerak. Izoh dasturi boshqacha ifodalanmasligi va ma`lum bir kerakli ma`lumotlarga ega bo`lishi kerak. Izohlar to`g`ri bo`lib, dstur o`zgarganda o`zgaruvchan bo`lishi kerak. Dasturdagi noto`g`ri izohlar ularni yo`qligiga nasbatan ancha yomonroqdir.

Qatorlarni tashlab ketish - ko`pincha yaxshi baholanmaydigan, lekin dasturni ko`rinishini yaxshilaydigan metoddir. 1 ta qatorni tashlaganda mantiqan bog`langan operatorlarni har birini ajratamiz. 2 ta qatorni tashlaganda dasturni asosiy mantiqiy qismlarini ajratib olishimiz mumkin bo`ladi. To`ldirmagan qatorlardan foydalanib, dasturdagi alohida bo`laklarni aniq topilishini yengillashtirish.

Oraliq (Probel). Dasturlash tilida oraliqlar ixtiyoriy ravishda qo`yiladi. Oraliqlardan keng foydalanish dasturni o`qishni osonlashtiradi.

Tenglashtirish va ketma - ketlikni belgilash.

Dasturni ketma - ketligini belgilashda biz undagi operatorlarni ketma - ketligi tartiblarida sodir bo`ladigan xatoliklardan forig` bo`lamiz. Dasturni boshlang`ich qismidan belgilab borishda, bizga uni tekshirishimizda va sozlashimizda katta yordam beradi. CHunki tartib belgi yoki sonlar bizga katta dasturda kerakli qatorlarni aniqlashimizda yordam beradi. Tenglashtirilgan va ketma - ketligi belgilangan dasturlar chiroyli va o`qishga qulay bo`ladi.

O`zgaruvchilarga nom tanlash.

O`zgaruvchilarni nomini shunday tanlash kerakki, bu qismda o`zgaruvchilarni kattaliklari yaqqol bilinib turishi kerak. Dasturni o`qishi oson kechishi uchun ismni to`g`ri tanlashga ham bog`liq bo`ladi. Butun o`zgaruvchilarning nomi I, G, K, L, M, N harflardan boshlanadi.

O`zgaruvchilar quyidagi turlarga bo`linadi: butun (tselo`e), haqiqiy (deystve), kompleksli, belgili (simvolg`no`e).

Fayllar nomi.

Fayllar bilan ishlayotgan vaqtda ularni solishtirish uchun qandaydir perefiks yoki suffiks tanlanadi. Agar har bir fayl o`zining perfiksiga ega bo`lsa, dasturni o`qish oson bo`ladi. Bu perfiks dasturni yozishda mos keluvchi maydonni aniqlab beradi va qaysi ishchi bilan maydonlar mantiqan bog`langanligini aniqlab beradi.
Dasturlarni har hil dasturdagi bir hil fayllarga bir hil nom berilishlari, shu dasturlarni o`zaro solishtirish jarayonlarini osonlashtiradi.

Standart qisqartirishlar.

Dasturni ishlab chiqishda standart qisqartirishlarni kiritish shartdir. CHunki bo`lar begona shaxslarni dasturingizni o`qishini osonlashtiradi. Standart qisqartirishlar dasturchiga kerak bo`lganda eski dasturlarni yangisini yozishga yordamlashadi. Standart qisqartirishlarni ishlab chiqish quyidagilardan iborat: har bir so`zning belgisi xotiraga olinishi kerak. Saqlangan ismdagi so`zlarning umumiy soni 3 dan ortmasligi kerak. Standart qisqartirishlarda har doim so`zlarning bosh harfi kiritilishi shart. Undosh harflar unli harflarga nisbatan loyihalashliroq. So`zning boshi uning oxiridan ko`ra kerakliroq. Standart qisqartirishlar o`z ichiga 6 dan 15 gacha harflar olishi kerak.

Keyingi satrga ko`chirish (Perenos).

Agar siz yozayotgan qatorga biron bir so`z sig`may qolsa, uni keyingi qatordan boshlang. Albatta ko`chirish ruxsat etiladi. Lekin u dasturni o`qishni va undan foydalanishni qiyinlashtiradi.

Operatorlarni joylashtirish. Bir xilgi dasturlarda bitta qatorga bir necha operatorlarni joylashtirish mumkin. Lekin bu 2 ta sabablarga noqulay: birinchidan, dasturni o`qish qiyinlashadi. Ikkinchidan, u paragraflarga bo`lish kabi uslubdan foydalanishni almashtiradi. YAna bir sabablardan biri shundaki, sintaksis xato haqida xabar borayotgan vaqtda, qatorning nomeri ham beriladi.

Alfavit bo`yicha ro`yxatni tartiblash.

Dasturlash tilida juda ko`p o`zgaruvchilar qatnashadi va ularni ismi bo`yicha aniqlash tuzuvchining o`ziga bog`liqdir. SHuning uchun tuzuvchining ishini osonlashtirish uchun ro`yxatlar tuziladi va ro`yxatlar 2 ta turga bo`linadi:

1. O`zgaruvchilar turida eolon qilingan o`zgaruvchilar nomi ro`yxati.

2. Jarayon parametrlarining ro`yxati.

Alfavit bo`yicha tartiblashni operator argumentlari ro`yxatida jarayonlarni chaqirib tartiblash ham mumkin. Kichik ma`lumotlarni tartiblashning boshqa usuli - bu belgilarni nomerlashdir. Ro`yxatlar ustun shaklida bo`lishi kerak. Dasturchi ro`yxatni ustun shaklida tuzish uchun har bir o`zgaruvchilar ro`yxatidagi ismlarning eng ko`p belgilariga mos ravishda ismlar ostida joy qoldirish shart. Bunday joylashtirishlar kirish - chiqishdan tortib har bir berilganlar ro`yxatida bo`lishi kerak.

Qavslar.

Qavslardan to`g`ri foydalanish dasturni o`qishni osonlashtiradi, chunki operatsiyalar bajarilayotgan vaqtda ma`lum bo`ladi va bunda dasturchi bir necha qavslardan foydalanishi mumkin, lekin bunda dasturni o`qish va tuzatish qiyinlashadi.

Asosiy qoidalardan biron - gumonli joylarda qavs qo`ying. Bu esa dasturni tushunarli qilib, xato qilishni oldini oladi.

Qator boshidan o`tish.

Operatorlarni yozishda ularni o`rtasidagi bog`liqni ko`rsatish uchun bir satrlar boshidan o`tishlar bajariladi. To`g`ri bajarilgan o`tishlar dasturni tarkibini ko`rsatadi. Bu matnni tushunarli tilda paragrflarga guruhiy so`zlarni mantiqiy bog`lab bo`ladi. tsikllar - o`tishlarda foydalanishdagi oddiy holat. O`tishlarda faqat tsikllarda emas, balki buyruqlarni qulay guruhlashda ham kiritiladi.

Nazorat savollari:

1. Quyidagi terminlarga izoh bering:
2. a) kirish izohlari; v) tushintiruvchi izohlar s) sarlavxali (mundarija)
3. Nima uchun programmalar qulay o`qiladigan bo`lishlari kerak?
4. Izohlarni tez-tez berilishi shartmi?
5. Programmada bo`sh qator nima maqsadda ishlatiladi?

6. Fayllarni nomini ko`rsatishda nimaga eotibor berish kerak?
7. Kirish izohlari nechta elementlardan iborat bo`lishi shart?
8. Programma tuzishda qator boshidan tashlab yozish qaysi vaqtlarda amalga oshiriladi.
9. Programmada standart qisqartirishlarni qo`llanish afzalligi nimada?

2 - Mavzu. Programmalashtirish yo`li. Bo`limlar nomini tanlash. "O`qilmaydigan" programmalar. Programmalovchiga maslaxatlar.

R E J A :

1. **Bo`limlar nomini tanlash.**
2. **"O`qilmaydigan" programmalar.**
3. **Programmalovchiga masalaxatlar.**

**Tayanch so`zlar:** *Bo`limlar nomini tanlash. "O`qilmaydigan " programmalar. Programmalovchiga maslaxatlar, bo`sh xonalar nomlash, tartibli nomerlash. O`zgaruvchilarning nomini tanlash, fayllar nomi, qisqartirishlar, operatorlarni joylash.*

Bo`limlarning nomini tanlash.

Bo`limlarning nomi shu bo`limning maqsadini bildirib turishi kerak. Bundan tashqari, bo`limlardan foydalanishda izohlarga qulay joy ajratib beriladi. har bir bo`limning boshida shu bo`lim haqida izoh joylashtirish kerak.

Dasturlash usullari dasturni o`qilishini qulayligi bilan o`zaro bog`liq. Usullar deganda tajribali dastur tuzuvchilar to`g`ri natijali, foydalanishda va o`qishda qulay bo`lgan dasturlarni tuzishda foydalanadigan usul va metodlar nazarda tutiladi.

Dasturlash usullari dasturni o`qilishini qulayligi bilan o`zaro bog`liq. Usullar deganda tajribali dastur tuzuvchilar to`g`ri natijali, foydalanishda qulay bo`lgan dasturlarni tuzishda foydalanadigan usul va metodlar nazarda tutiladi.

Dastur usulining eng yaxshi qoidasi - bu tajribali dasturchilar o`rtasidagi kelishishlardir. CHunki bitta tajribali dasturchiga nisbatan 2 ta yoki 3 ta tajribali dasturchilar tuzgan dastur yaxshiroqdir.

Dastur shunday tuzilishi kerakki, uni birinchi navbatda mashina yoki EHM emas, balki inson o`qib tushunsin. CHunki dastur insonlarga dasturlarni tuzishda, shakllantirishda va ularni qo`llashda kerak bo`ladi.

Dastur - bu keyinchalik ishlatishga va takomillashtirishga mo`ljallangan hujjat, algoritmlarni kodlashtirish uchun o`quv materiali va hokazolardir.

Demak, dasturlash tillari bizga o`qishda qulay bo`lgan dasturni yaratishni ta`minlab berishi kerak. Dastur iloji boricha algoritimning tuzilishini va mantig`ini bizga tushunarliroq qilib yetkazib berishi kerak.

Usullarni standartlash.

Albatta biz juda ko`p joylarda va jarayonlarda standartlarga asosan ish olib boramiz. Lekin dasturlash usullarida bunga qarshi bir necha fikrlar aytilgan. Ulardan birini keltirib o`tamiz: dasturlash usuli - bu shaxsiy fikr va didga bog`liqdir. SHuning uchun unga hech qanday chegaralanishlar qo`yilmasligi kerak.

Usullarni standartlash quyidagilardan iborat:

Agar biron - bir narsani yaratishda bir necha usul mavjud bo`lib, tanlash sizga bog`liq bo`lsa, unda siz 1 ta usulni tanlab, oxirigacha shu usuldan foydalaning.

Ammo standartlash jarayonining o`ziga xos kamchiliklari bor:
standartlashni qo`llaganda kelajakda dasturlash yo`llarini sekinlashtiribgina qolmay, balki foydalanishda qulay va ortiqcha chegaralarga ega bo`lishi mumkin. CHunki ular yangidan - yangi vazifalarni bajarishga sizning aqlingizni bog`lab qo`yadi.

"O`qilmaydigan " programmalar.

Dastur ishlab chiqilgandan so`ng uni va javobgar shaxs, bevosita loyihalovchi tasdiqlaydi. Aniqlikka talab standart usuli so`rab o`tirilmaydi. Faqat ishlaydimi? ishlamasa qachon ishlaydi? Bunday dasturlar kirayotgan ma`lumotni qabul qiladi va chiqaruvchi ma`lumotni chiqaradi. Xuddi “oldim, berdim”. SHuning uchun bu dasturlar katta o`zgarishlar sodir bo`lgunga qadar kerakdir. Bu o`zgarishlardan so`ng bunday dasturlar tashlab yuborilib yangilari paydo bo`ladi. CHunki eskisini

takomillashtirishdan ko`ra yangisini loyihalash osonroq. SHuning uchun bunday dasturlarga ikkinchi shaxslar qo`l qo`yishi va dasturning usuli va o`lchami, hamda ishlashiga javob berishlari maqsadga muvofiqdir.

Qavslar. (Skobki).

Qavslardan to`g`ri foydalanish dasturni o`qishni osonlashtiradi, chunki operatsiyalar bajarilayotgan vaqtda priariatatsiya ma`lum bo`ladi va bunda dasturchi bir necha qavslardan foydalanishi mumkin, lekin bunda dasturni o`qish va tuzatish qiyinlashadi.

Asosiy qoidalardan biron - gumonli joylarda qavs qo`ying. Bu esa dasturni tushunarli qilib, xato qilishni oldini oladi.

Qator boshidan o`tish.

Operatorlarni yozishda ularni o`rtasidagi bog`liqni ko`rsatish uchun bir satrlar boshidan o`tishlar bajariladi. To`g`ri bajarilgan o`tishlar dasturni tarkibini ko`rsatadi. Bu tekstni tushunarli tilda paragrflarga guruhiy so`zlarni mantiqiy bog`lab bo`ladi. tsikllar - o`tishlarida foydalanishdagi oddiy holat. O`tishlarda faqat tsikllarda emas, balki buyruqlarni qulay guruhlashda ham kiritiladi.

Operator IF yordamida aniqlanadigan operatorlarga qator boshidan bir o`tishlar bilan yoziladi. Kiritish - chiqarish operatoriga yozishda ham o`tishlardan foydalaniladi. teng xuquqli fayllarni taoriflashda ham o`tishlardan foydalaning.

Agar dasturchi tuzayotgan dasturni biron - bir boshqa dasturchi yoki kishini o`qishini bilsa, u dasturni kodlashdagi fikri keskin o`zgaradi. U dasturni programmalarga bo`lib, izohlar bilan to`ldiradi. Natijada dastur aniq va tushunarli bo`ladi.

Bir xilgi loyihalarda dasturni oddiyligini ta`minlovchi metod dasturni nazaorat qiluvchi sistemadan foydalaniladi.

Bu sistemada dastur loyihalanaetgan paytda kamida 2 ta dasturchidan foydalaniladi. Birinchisi dastur to`zsa. ikkinchisi shu dasturni tuzish uchun ko`rib, o`qib chiqadi. Bu sistemani 2 ta yutug`i bor. Birinchidan u oddiy va tushunarli dastur ishlab chiqishni ta`minlaydi. Ikkinchidan, birinchi dasturchi loyihani tugata olmasa, ikkinchi dasturchi uni o`rniga tugatib qo`yadi.

Dasturni standartlarini bajarishlikka baholash metodi ham shunga o`xshash metoddir.

Dasturchiga maslahatlar.

- Keragidan ko`proq va yana ko`proq izoh bering.
- Kiritish izohlaridan foydalaning.
- Katta dasturlarda mundarija qiling.
- Izohlar boshini aylantirmasdan qo`shimcha ma`lumotlar bersin.
- Noto`g`ri izohdan ko`ra, uning yo`qligi yaxshi.
- Dasturni oson o`qilishi uchun oraliqlar bajaring.
- Fayllarni nomlashda prefiks va suffikslardan foydalaning.
- Alvafit buyicha ro`yxatlar tuzing.
- Dastur tarkibini aniqlash uchun o`tishlar xosil qiling.
- Ma`lumot tarkibini aniqlashda ham erinmang. Qator boshidan o`tishlardan foydalaning.

Nazorat savollari:

1. Kirish izohlari nechta elementlardan iborat bo`lishi shart?
2. Programma tuzishda qator boshidan tashlab yozish qaysi vaqtlarda amalga oshiriladi.
3. Programmada standart qisqartarishlarni qo`llanish afzalligi?
4. Nima uchun programmalar qulay o`qiladigan bo`lishlari kerak?
5. Izohlarni tez-tez berilishi shartmi?
6. Programmada bo`sh qatorli nima maqsadda ishlatiladi?
7. Fayllarni nomini ko`rsatishda nimaga eotibor berish kerak?
8. Programma tuzishda qator boshidan tashlab yozish qaysi vaqtlarda amalga oshiriladi.

3 - Mavzu. Programmalarni loyhalashtirish. Masalani tavsiflash, algoritmni tanlash, ma`lumotlarni tavsiflash, programmalash tilini tanlash.

R E J A :

1. **Masalani tavsiflash.**
2. **Algoritimni tanlash.**
3. **Ma`lumotlarni tavsiflash.**
4. **Programmalash tilini tanlash.**

**Tayanch so`zlar:** *Masalani tavsiflash, algoritmni tanlash, ma`lumotlarni tavsiflash, programmalash tilini tanlash, bo`sh xonalar nomlash, tartibli nomerlash. O`zgaruvchilarning nomini tanlash, fayllar nomi, qisqartirishlar, operatorlarni joylash.*

Dasturni loyihalash bosqichi dasturlash usuliga, ishonchligiga, sozlashga, foydalanish tarkibiga, va xokazolarga ta`sir ko`rsatadi.

Dasturning loyi`alashning oson kechishi - bu dasturni oson o`qilishi uchun qo`yilgan birinchi qadam. Kodlashtirish oddiyroq bo`lsin.

Ortiqcha shakllantirilgan dastur sozlashda yoki modifikatsiyalashda kimmatga tushadi. Dasturdan faqat uning muallifi emas balki boshqa shaxslar ham foydalanib, ham sozlashda kiynalmasligi kerak. Dasturning tarkibi uning mantig`ini ko`rsatishi shart, kodlarning doimiyligi dasturni yanada osonlashtiradi. Masalan: dasturni o`zgartiruvchilar doimo bir xil ravishda foydalanish kerak. "1" belgisi yoqilgan. "0" belgisi o`chirilgan. Bundan tashqari, dasturdagi har bir jadval bir xil taosvirlansin.

Masalani qo`yilishi va tarkibi.

Loyihalash vaqtida har - xil muammolarni yuzaga kelishini hisobga olish kerak. Asosan bu muammolar shu dasturni buyurtmachilarining aniq maqsadga ega bo`lmasliklaridan kelib chikadi. Foydalanuvchi noaniq bergan so`rovga aniq va to`g`ri javob beruvchi dastur - bu eng yuksak dasturdir. Vazifa yaxshi yoki yomon deb aniqlanishi mumkin. YOmon shakllantirilgan vazifaga aniq bir dasturni loyihalab bo`lmaydi.

Aniq belgilangan vazifa esa loyihalash davomida ortiqcha ishdan, vaqt sarflashdan ozod bo`ladi. Vazifa dasturning xajmiga qarab ixcham va tushunarli bo`lishi kerak. Agar dastur kichkina bo`lsa vazifa bir necha qatordan, katta bo`lsa butun - bir kitobdan iborat bo`lishi mumkin. Bu tavsiflar katta songa ega bo`lgan hujjatlarga asos bo`ladi. Keyinchalik vazifalarga kiritilgan o`zgartirishlar yoki qo`shimcha vazifalar dasturni bo`g`ibgina qolmay uni tugatishni umuman sekinlatib qo`yadi. Vazifani aniqlashda qo`yilgan loqaydsizliklar keyinchalik katta muammolar keltirib chiqaradi.

Algoritm tanlash.

Algoritmni to`g`ri tanlash bu dasturni to`g`ri va samarali bo`lishiga quyiladigan asosiy qoidadir. Bunda dasturni tilini va tarkibini to`g`ri tanlash mumkin. SHunday qilib yaxshi algoritm kerakli, lekin dastur uchun yetarli sharoit emas.

Birinchidan - miyaga to`g`ri kelgan algoritmga dastur tuzmaslik kerak. Bir necha variantlarni ko`rib chiqib, ularning ichidan eng yaxshisini tanlash zarur.

Ma`lumotlarni tavsiflash.

Yaxshi tavsiflangan ma`lumotlar dasturni ancha qisqartiradi. Demak, ma`lumotlar massivni tashkillashda zarur bo`lgandagina qo`llash kerak. Boshqacha usulida tayanch va ko`rsatkichlardan foydalanish kerak. Ma`lumotlar haqida ko`rilayotgan vazifaga mos holda taosurotga ega bo`lish kerak.

Hisobda har bir ma`lumotning tarkibini hamma tilga ko`chirish mumkin. Lekin siz mo`ljallangan ma`lumot tarkibini mujassamlagan dasturlash tilida foydalanishingiz ma`qulroq.

yetarli murakkablikdagi strukturasi modullarini tuzish.

Programmani modullarga bo`lish va yuqoridan pastga loyihalash murakkab strukturalarni nazorat qilish imkoniyatini berib, uni darajasini belgilaydi. Modullarga bo`lish - bu murakkablik darajasini boshqarish vositasidir. Agar pastgi qismlaridagi informatsiyalarga talab bo`lmasa ular qoldirilishi mumkin.

Birinchidan modullar o`zgaruvchilar to`plamini bo`lishga sharoit yaratadi. Buning okibatida modullar zarur bo`lmagan o`zgaruvchilardan xalos bo`lib, ko`zatilishi mumkin bo`lgan xatoliklardan ma`lum darajada himoyalanadi. Bundan tashqari, modullar qanchalik katta bo`lsa, uni taxlil qilish shunchalik qiyin. SHunday qilib modullarni kiritilishi programmada bo`lishi mumkin bo`lgan yo`llar sonini kamaytiradi va uni boshqarishni yengillashtiradi.

Programmalash tilini tanlash.

Ko`pincha dasturlash tilini hisoblash sistemasi yoki dasturchi foydalanuvchiga asosan tanlangan bo`ladi.

Agar dasturchi tanlashga ega bo`lib qolsa u albatda vazifaga mos eng yuqori dasturlash tilini tanlashi kerak. Agar tanlangan dasturlash tili berilgan vazifaga mos kelmasa dasturlashda va sozlashda muammolar kelib chiqishi mumkin. Tanlangan til loyihalashga ham ancha taosir ko`rsatadi.

Nazorat savollari:

1. Dasturni loyihalashtirishga tushuncha bering?
2. SHEHM da axborot qanday tartibda kiritiladi?
3. Loyihalashda masalani qo`yilishi qanday loyihalashga ega?
4. Algoritmni tanlash, programmalash tilini tanlash dasturni loyihalashtirishda qanday loyihalashga ega?
5. Nima uchun programmalar qulay o`qiladigan bo`lishlari kerak?
6. Izohlarni tez-tez berilishi shartmi?
7. Programmada bo`sh qatorli nima maqsadda ishlatiladi?
8. Fayllarni nomini ko`rsatishda nimaga e`tibor berish kerak?

9. Kirish izohlari nechta elementlardan iborat bo`lishi shart?
10. Programma tuzishda qator boshidan tashlab yozish qaysi vaqtlarda amalga oshiriladi.

4 - Mavzu. Programmalarni loyihalashtirish. Universallik, kutubxonalar, kiritish va chiqarish formatlari, maqsadni aniqlash, murakkablik.

R E J A :

1. **Universallik.**
2. **Kutubxonalar.**
3. **Kiritish va chiqarish formatlari.**
4. **Maqsadni aniqlash. Murakkablik.**

**Tayanch so`zlar:** *Universallik, kutubxonalar, kiritish va chiqarish formatlari, maqsadni aniqlash, murakkablik, bo`sh xonalar nomlash, tartibli nomerlash. O`zgaruvchilarning nomini tanlash, fayllar nomi, qisqartirishlar, operatorlarni joylash.*

Universallik.

Universallik deb aniq ma`lumotlar turkumiga bog`liq bo`lmagan dasturlarni ataymiz.

Parametrlar sifatida kontaktlar o`rniga o`zgaruvchilardan foydalaniladigan dasturni universal dastur deymiz. O`zgaruvchilarni kontak urnida qo`llanishi mumkin bo`lgan sharoitlarni ko`rsatib o`tamiz:

1. Ro`yxatlar, massivlar va tablitsalar o`lchami.
2. Fizik konstanta va protsentlar, oraliqlar.
3. "Kiritish - chiqarish" moslamasini belgilanishi.

Dasturlashni universalligi loyihalashda vaqtdan yutishi va undan foydalanishiida keng sharoitlar ochib berishdir.

Universal protseduralar dastur biblioteekasiga keyingi das-
turlarda ham foydalanish uchun kiritiladi.

Universallikni qidirishda asosan keyinchalik qo`llaniladigan modifikatsiya va variantlarni aniqlashda ham yordam beradi.

Kutubxonalar.

Dasturni samaradorligini oshirish, sozlash va tekshirishni yengillashtirsh hamda dastur yaratishni qisqartirish uchun dastur kutubxonalari qo`llaniladi. Kutubxona uchun mo`lljallangan kichik dasturlar katigoriyasiga dasturlash tiliga ega bo`lgan funktsiya yoki kichik dasturlarni o`z ichiga oladi. Bu katigoriyalarga barcha standart funktsiyalar (sin, cos va xakazo), kiradi.

Ikkinchi funktsiya va kichik dasturlar manbaisi sizning hisoblash sistemangizdan tashqarida joylashgan. Foydalunuvchiga u yoki bu sistemada bajarilgan dasturlar to`g`ri keladi. Kutubxona dasturlarda foydalanish sabablaridan biri ishonchlikni oshirish va dasturlashdagi qiyinchiliklarni kamaytirishdir. kutubxona dasturlarining uchinchi manbaisi bu dasturlar va hisoblash mashinalari to`g`risidagi kitoblardir.

"Kiritish - chiqarish" formatlari.

Ma`lumotlarning "kiritish - chiqarish" formatlari loyihalash etapining bir qismi hisoblanadi. Kiritish formatlari foydalanuvchi uchun eng yuqori qulaylik va eng kam xatoliklar bo`lishini hisobga olgan holda bajariladi. CHiqish ma`lumoti tashqi manbalar ta`sirisiz solishtirilishi kerak. U o`z ichiga:

1. CHiqish yozuvini solishtirish.
2. YOzuv funktsiyasi yoki tavsifi.
3. Sana.
4. Varaklarning tartib raqamini olishi kerak.

Yaxshi o`ylab topilgan chiqish formatlari tuzuvchiga sozlashda va sinashda katta yordam beradi.

Maqsadni aniqlash.

Loyi`alash vaqtida dasturchi o`z oldiga ma`lum bir aniq maqsadlar qo`yishi kerak. Quyida ularning bir nechasi keltirilgan.

1. Eng yuqori ishonchlilik.
2. Ma`lum bir ishning ayrim xajmini aniq bir sanaga bajarish.
3. Loyihalash uchun eng kam vaqt sarflash yoki tan - narxni arzonligi.
4. Foydalanishdagi qulayligi va oddiyligi.
5. Samaradorligi (xotira xajmi yoki tez ishlashi).
6. Takommilashtirishni hisobga olish.
7. Universallik.

Ayrim xollarda bitta dastur ustidan ishlayotgan ikkita dasturchi har xil maqsadlarni ko`zlab ishlashadi.

Biri xotira xajmini kichkaytirishga harakat qilsa, ikkinchisi loyihalash uchun sarf - harajatni kamaytirishni hisoblaydi. Maqsadlarni oldindan va aniq qo`ying.

Dasturiy ta`minlash sistemasidan foydalanish vaqtida yangidan yangi maqsadlar yuzaga keladi. YAngi maqsadlarga asosan tayyor dasturni o`zgartirish - pog`onaga (tiqishtirish) deb ataladi.

Dasturga yangi funktsiyalar va eskisini yangilashda pog`onaga kul-
laniladi.

Murakkablik.

Hammaga ma`lumki, dasturda 2 ta asosiy parametrlar: xotira va sanalari bor.

YAna murakkablik parametrini ham hisobga olish kerak. Dastur qanchalik murakkab bo`lsa, uni tekshirish, nazorat qilish, sinash, tartiblash shunchalik qiyin bo`ladi. Murakkablik parametrlarini chegaralaganimizda biz boshqarishni ancha yengillashtiramiz. Xozirgi vaqtda murakkablikni chegaralovchi usullar va metodlar bor.

Murakablikni boshqarish metodlari nima? Ular qanday bajaradi? Dasturlashda ularni qanday qo`llaniladi.

Sifatni oshirishning 2 ta yo`li bor: biri kattik va chuqur tekshirishlar o`tkazish, ikkinchisi har bir loyihalash jarayonida sistemalarga eng yuqori ishonchlilik bilan ta`minlash.

Murakkablikni boshqarish metodida asosan jarayon yoki tarkib bir necha parametrlarga bo`linadi va aniq bir funktsiyani bajarish uchun foydalaniladi. Bunday metodlar barcha ishlab chiqarishda, administrativ ishlarda va tashkilotlarda foydalaniadi. Murakkablikni boshqarish dasturlashdagi asosiy funktsiyalardan biridir.

Nazorat savollari:

1. Katta xajmdagi dasturlarni nima uchun programmalashtirish murakkabroq?
2. Unversallik deganda nimani tushinasiz?
3. Kutubxonalarni dasturni loyihalashda vazifasi?
4. “Kiritish- chiqarish” formatlariga tushuncha bering?
5. Dasturni loyihalashtirishga tushuncha bering?
6. SHEHM da axborot qanday tartibda kiritiladi?
7. Loyihalashda masalani qo`yilishi qanday loyihalashga ega?
8. Algoritmni tanlash, programmalash tilini tanlash dasturni loyihalashtirishda qanday loyihalashga ega?
9. Sarlavxali (mundarija)
10. Nima uchun programmalar qulay o`qiladigan bo`lishlari kerak?

5 – Mavzu. Strukturali programmalash. Kichik xajmdagi programmalarni ishlab chiqarish usullari. Katta xajmdagi programma loyihalarni tadbiq qilish yo`llari. Programmalar strategiyasi: YUqoridan pastga jarayoni.

R E J A :

1. **Kichik xajmdagi programmalarni ishlab chiqarish usullari.**
2. **Katta xajmdagi programma loyihalarini tadbiq qilish yo`llari.**
3. **Programmalar strategiyasi: YUqoridan pastga jarayon.**

**Tayanch so`zlar***: Kichik xajmdagi programmalarni ishlab chiqarish usullari. Katta xajmdagi programma loyihalarni tadbiq qilish yo`llari. Programmalar strategiyasi: YUqoridan pastga jarayoni.*

Programmam tuzishninig eng asosiy usullaridan bir bu strukturaliy programma tuzishdir. Bu usulda programma tuzish uchun uchta qism mavjud:

1. YUqoridan pastga programma tuzish.
2. Modul programmalashtirish.
3. Strukturali kodlash.

YUqoridan pastga programma tuzishda programmaning yuqori qismdan boshlanadi. Programmaning asosiy qismi tuzilib, quyi qismidagi modullar esa vaqtinchalik fakt nomlari bilan atalgan proseduralar bilan almashtiriladi. Programmani asosiy moduli tuzilib, testidan o`tkazilgandan so`ng ketma-ket vaqtincha yozilgan modullarni yozish bilan programma tuzish davom ettiriladi.

Modul prolgrammalashtirishda programmani logik qisimlariga bo`linadi. Bu modullar programmada protseduralar va funktsiyalar orqali amalga oshiriladi.

Strukturali kodlash deganda har bir modulni gorizantal va vertikal qatorlarda to`g`ri nomlanishiga aytiladi. Bu usul yordamida modullardan tuzilgan programmalar

ishlaydigan testidan o`tkazishi qulay modifikattsiya qilish uchun qulay programmalar yaratish mumkun.

Programma tuzatishning texnologik jarayoni.

Programma tuzish quyidagi etaplardan amalga oshiriladi:

-Vazifani qo`yilishi. Bu etapda programmist buyurtmachi yordamida yechilishi kerak bo`lgan vazifani qo`yadi. Texnik topshiriq tuziladi.

Bunda programmaning asosiy harakteristikalari, muddatlar va ma`sul shaxslar aniqlanadi.

-Algaritmni tuzish. Programmist vazifani taxlil qilib kerakli bo`lgan algaritmni tanlaydi. Tanlangan algaritm to`liq taxlil qilinadi va uning blok-sxemasi chiziladi.

-Programmalashtirish etapi. Dastur yaratish tili tanlanadi.

Programma qabul qilingan algaritimda tuziladi.

-Programmani tuzatish etapi.

-Programmani testidan o`tkazish etapi.

Odatda dasturni testdan o`tkazishda etaplarga bo`lib o`rganila-

di. Bunda `ar bir modulni tekshirishdan tortib, to butun sistemani yakuniy tekshirishlar kabi etaplarni oladi. Agar bunda biron bir ishonchli ketma - ketlikka yondashilmasa, ishonchli ta`minlovchi dastur olish juda qiyindir. Testlash strategiyasi ikkita usuldan birortasiga asosan bajariladi: odatiy quyidan - yuqoriga qarab testlash, yoki zamonaviy yuqoridan - pastga qarab testlash.

Quyidan - yuqoriga qarab testlash.

Bu usul keng tarqalgan usul bo`lib, unda eng quyi pog`onadagi

boshlangich yozilgan modullar tekshiriladi. So`ngra yuqori qatlamdagi elementlar dasturlanadi va testlanadi. Bu jarayon to yozilgan dastur butunlay yakunlanmaguncha davom etadi. Quyidan - yuqoriga qarab testlash usuli xozirgi vaqtda yuqoridan - pastga qarab testlovchi va dasturlovchilar tomonidan qo`lanmayapti. Ularni fikricha bu usulda interfeys va algoritmdagi ko`pgina xatolar aniqlanmay qolib ketmoqda. Bu esa dasturni qayta-qayta o`zgartirishdan so`ng buzishga olib keladi.

Ikkinchi kamchiligi esa: har xil pog`onadagi elementlarni testdan o`tkazishda yangidan yangi testlovchi moslamalarni, drayverlarni va testlovchi ma`lumotlarni talab qilmoqda. Bu esa o`z-o`zigadan dasturlashda katta xajmda mehnat talab qiladi.

YUqoridan pastga jarayoni.

Bu testlash usuli yuqoridan pastga qarab dasturlashni, yuqoridan pastga qarab kodlashni qo`shimcha etapi hisoblanadi. Bu usulda oldin asosiy dastur yoziladi va so`ngra past pog`onadagi loyihalanmagan elementlar urin bosuvchi dasturlar bilan almashtiriladi. Bunday skeletli dastur chaqiriluvchi dastur va har qanday malumotlar yo`qligida ham o`z ishini davom ettiradi.Bu tekshirish natijasida bazi xollarda bemani bo`lgan xatolar ham aniqlanadi. Keyingi qadam modul kushilishidan iborat bo`lib,unda bu modullar kiruvchi modullarni ko`paytiruvchi bo`lishi ham mumkin, - bu esa kiritish moduli,bazi bir yordamchi modul (oxirgini dasturlash tugash daqiqasiga qadar) bo`lishi mumkin.Bu tekshirishdan so`ng sinash oddiy bir sodda kiruvchi ma`lumotlar bilan o`tkazish mumkin.

Nazorat savollari:

1. Strukturali programmalashtirishdagi qismlarni ko`rsating?
2. Loyihani qo`llash kutubxonasi (BPR- biblioteka podderjki razrabotki) ning maqsadini ko`rsating?
3. Modulli programmalashtirishni afzallligi nimada?
4. Dasturni loyihalashtirishga tushuncha bering?
5. SHEHM da axborot qanday tartibda kiritiladi?
6. Loyihalashda masalani qo`yilishi qanday loyihalashga ega?
7. Algoritmni tanlash, programmalash tilini tanlash dasturni loyihalashtirishda qanday loyihalashga ega?
8. Sarlavxali (mundarija)
9. Nima uchun programmalar qulay o`qiladigan bo`lishlari kerak?
10. Izohlarni tez-tez berilishi shartmi?

6-Mavzu. Strukturali programmalashtirish. Amaliy programmalashtirish.

R E J A :

1. **Amaliy programmalashtirish.**
2. **Bosh dasturchining brigadasi.**

Tayanch so`zlar: Amaliy programmalashtirish. Bosh dasturchining brigadasi.

Amaliy programmalash ham yuqoridan quyiga qarab bajariladi, bunda programma boshida topshiriuklarni blshqaruvchi tillar operatorlari yoziladi, undan keyin asosiy boshqaruvchi programma yoziladi.

Odatda loyiha "skeleti" bitta odam tomonidan bajariladi, bu ishlab chiqilgan loyihani ta`minlaydi. Modullik printsipi ishlatilganligi uchun bosh programma qisqa bo`lishi, shuningdek modul va quyi programmalarni chaqirib bajariladigan bo`lishi kerak, buni quyi programmalarni yaratish yordamida amalga oshirish mumkin.

Ikkita loyihani qayta ishlash vaqtida strukturali programmalashtirishning hamma metodlari birlashtirilib, bosh dasturchini brigadasi deb nomlanuvchi brigada yaratiladi.

Bosh dasturchini asosiy majburiyatlari quyidagilardan iborat:

1) Texnik boshqarishda-loyiha texnik aspektini kuzatish;

2) Xodimlarni boshqarishda-odam sonini nazorat qilish;

3)Kontrakt shartlarni bajarishda-mijozlar bilan muloqotni bosh- qarish.

Qayta ishlashni tayanch kutubxonasi (KITK)-Hamma strukturali programmalarni saqlash va ishlatish uchun hizmat qiladi, bu programmalar testdan utgan va sistemaga qo`shilgan bo`ladi. KITK quyidagilar uchun mo`ljallangan:

1. Loyiha joriy xolatini aniqlashni tashkil qilish va saqlash hamda hohlagan vaqtda undan foydalanishni tashkil qilish.

2. Loyiha xodimlarni qolgan ishdan ozod qilishni tashkil qilish.

Bunday tashkilot programmalshni unumdorligini oshiradi.

KITK quyidagi xususiyatlrga ega bo`lishi kerak:

1. Mashinaning ichki va tashqi hujjatlarini saqlash kerak;

2. Butun loyiha davomida ishlashi kerak. Loyihalash bosqichidan to oxirgi yakunlovchi past-sinov bosqichigacha va loyiha buyicha hisobot tayyorlashgacha

3. Tashqi kutubxona faktgina eski tarixni saqlabgina qolmay, balki loyihaning joriy xolatini ham saqlaydigan bo`lishi kerak.

4. Prrogrammalar o`zlari berilgan formatlari, programma ishlashi, algoritmlar va xokazo haqida savol tugilgagnda ma`lumot berishlari kerak.

5. Iloji boricha kutubxonachi o`zi hamma "progon" larni bajarishi va registratsiya qilishi kerak. Bu loyiha butunligini va nazorat ostida bo`lishini ta`minlaydi.

Arxiv bo`lishi bir nechta sabablarga ko`ra foydalidir. Bunda programma evolyutsiyasini kurish imkoniyati yoki bu programmaning oldingi versiyalarini qurish imkoniyati yaratiladi.

KITK ning bitta qulayliklaridan biri shuki, bunda xoxlagan faylni tiklash imkoniyati borligidir.

Nazorat savollari:

1. Amaliyot programmalashtirishga tushuncha bering?
2. Bosh dasturchini brigadasining vazifasi

3. Birigada a`zolarining vazifalari?
4. Modulli programmalashtirishni afzallligi nimada?
5. Dasturni loyi`alashtirishga tushuncha bering?
6. SHEHM da axborot qanday tartibda kiritiladi?
7. Loyihalashda masalani qo`yilishi qanday loyihalashga ega?
8. Algoritmni tanlash, programmalash tilini tanlash dasturni loyihalashtirishda qanday loyihalashga ega?
9. Nima uchun programmalar qulay o`qiladigan bo`lishlari kerak?
10. Izohlarni tez-tez berilishi shartmi?

7 – Mavzu. Programmaning samaradorligi. Samaradorlik. Programmalarni optimallashtirish. Kompilatsiya jarayonida optimizatsiyalash.

R E J A :

1. **Samaradorlik.**
2. **Programmalarni optimallashtirish.**
3. **Kompilatsiya jarayonida optimizatsiyalash.**

**Tayanch so`zlar:** *Samaradorlik. Programmalarni optimallashtirish. Kompilatsiya jarayonida optimizatsiyalash.Kichik xajmdagi programmalarni ishlab chiqarish usullari. Katta xajmdagi programma loyihalarni tadbiq qilish yo`llari.*

Dasturlash samaradorligi.

Dasturlashning asosiy vazifasi samarador dastur emas balki to`g`ri va ishonchli dastur yaratish. To`g`ri lekin samarasiz dasturni optimallab to`g`ri samarador dastur qilish mumkin. Lekin samarali, ammo noto`g`ri dastur ni kamdan kam xollarda to`g`rilash mumkin bo`ladi. SHuning uchun optimallash dasturlashning ikkinchi etapi hisoblanadi.

To`g`ri dastur olish esa birinchi etap bo`ladi. Noto`g`ri dasturiy ta`minlash ming u samarali bo`lsin baribir u foydasiz bo`ladi. CHunki uni ugnash ko`p tomonlama sarf etishni talab etadi.

Ko`pincha vaqtning ko`p qismi (uning xajmiga nisbatan %) dasturni uncha katta bo`lmagan kiritik deb nomlanuvchi qismni bajarilishga sarflanadi. Bilimlarni samadorligini orqasidan quvish suiste`mol qilishga olib keladi.

Samaradorlikka munosabat.

Dasturning 3 xili mavjud bo`lib, ularning har biriga samaradorlik ham turlicha bo`ladi:

Birinchi turiga ko`p foydadanuvchi dasturlar kiradi. Bularga jarayon samaralari, ko`ramalar (komplyatorlar), tadbiqiy dastur va adabietlarni hisobga oluvchi samaralar kiradi. Demak bo`lar uchun samaradorlikni ularni ko`p qo`llanilishi va o`ziga xos bajarilishi sabab asosiy vazifasidir.

Ikkinchi turiga o`zok vaqt ullaniladigan ishlab chiqarish dasturlari kiradi. Bu turdagi dasturlariniusta to`zatuvchilar tuzatadilar. Dasturlarni samaradorlikka jiddiy karalsada, ammoutgan dasturni ekspulatatsiyasiga katta loyihalash beriladi.

Uchinchi turga dasturchilar tomonidan emas, balki ilmiy xodimlar va administratorlar tomonidan tuzilgan dasturlar kiradi. Bu yerda samaradorlik dastur uchun tegishli bo`lib, berilgan xotira xajmida joylashgan va qo`llanilaetgan vaqtida bajarilishi kerak.

Dasturni samaradorligini oshirish ko`pincha usular uni oson o`qilishi uchun ta`sir ko`rsatadi. Dasturni oson o`qilishiga uning samaradorligiga nisbatan ancha jiddiy karaladi. CHunki bunday dasturlarni sozlash, modifikatsiyalash va undan foydalanish ancha osondir.

Optimallashtiruvchi kompyuterlar.

Samaradorlik dasturni loyihalashning ikkita bosqichida muximdir: Ko`ramalash va bajarish.

Ob`ekt dasturini samaradorligini yaratuvchi ko`rama juda katta xajmga ega bo`ladi.Lekin juda sekin ishlaydi. CHunki u ob`ekt dasturini optimallashtiradi.

SHu sababli xozirgi 1 ta mashina 2 tadan komplayatorlar har bir kirish tili uchun ishlaydi.

1 - chi komplyator tez ishlaydi, lekin samarasiz ob`ekt dasturlarini yaratadi. Bu komplyator dasturiy sozlashda ishlatiladi.

2 - chi komplyator sekin ishlaydi ammo dasturni optimllab samarali ishlab chiqaradi. Bu komplyatorlarda ob`ekt modullarini yaratilishiga foydalidir.

Samaradorlik.

Programani oz xajmi bilan ishlash samaradorlik deb ataladi. Lekin samaralorlik 2-etap bo`lib, birinchi etapda programmani to`g`ri ishlashini tashkil qilish kerak.

Inson faktorini `isobga olish.

Programmani foydalanuvchi ishlatganligi munosabati bilan inson faktorini hisobga olish zarur. SHuning uchun foydalanuvchining faktorlarini ham hisobga olish kerak. Progaramma "do`stona" bo`lishi kerak. Ekranlar uta tez almashishi, ma`lumotlar tez chiqib turishi ham insonga salbiy tahsir ko`rsatadi. Foydalanuvchi interfeysi (ekraniga) katta e`tibor berish kerak. Ekrandagi ranglar soni ham uncha ko`p bo`lmasligi kerak. Ranglar ham o`z o`rnida ishlatilishi kerak.
SHriftlar ham o`ta katta yoki o`ta kichik bo`lmasligi kerak.

Baxolanishi.

Programmani baxolash imkoniyati yaratilishi kerak.

Modifikatsiya kilinishi.

Asosiy harakteristkalaridan biri bo`lib, programmaga o`zgartirishlarni ishlab chiqarish jarayonida amalga oshirishga aytiladi. Programmani qayta yozmasdan qandaydir qismlarini o`zgartiriladi.

Ma`lumotlarni tafsiflash.

Yaxshi tafsiflangan ma`lumotlar dasturni ancha qisqartiradi, demak ma`lumotlar massivini tanlashda zarur bo`lganicha qo`llash kerak. Boshqacha usulda tayanch va ko`rsatkichlardan foydalanish kerak.Ma`lumotlar haqida qurilayotgan vazifaga mos holda taosurotga egabo`lish kerak. Hisobda har bir ma`lumot tarkibini hamma tilga o`zgartirish mumkin.Lekin siz mo`ljallagan ma`lumot tarkibini mujassamlagan dasturlash tilidan foydalanishingiz maoqulroq.

Dasturiy foydalanuvchilar ehtiyojlarini qondirish, keng tarqatish va sotish uchun mo`ljallangan.

Xozirgi paytda dasturiy maxsulotlarni ochiq (legal) tarqatishning boshqa variantlari ham mavjud, ular yalpi (global) va mintaqaviy kommunikatsiyalardan foydalanish bilan yuzaga keladi.

Nazorat savollari:

1. Programma tuzishda qaysi etapda programmaning samaradorligi muxim?
2. Qanday vaqtda programmani optimallashtirish kerak?
3. Programmani optimallashtirsh uchun qaysi usullardan foydalaniladi?
4. Programmani optimallashtirshda asosiy mezonlarni ko`rsating?
5. Modulli programmalashtirishni afzallligi nimada?
6. Dasturni loyihalashtirishga tushuncha bering?
7. SHEHM da axborot qanday tartibda kiritiladi?
8. Loyihalashda masalani qo`yilishi qanday loyihalashga ega?
9. Algoritmni tanlash, programmalash tilini tanlash dasturni loyihalashtirishda qanday loyihalashga ega?
10. Nima uchun programmalar qulay o`qiladigan bo`lishlari kerak?

8 - Mavzu. Programma samaradorligi. Ttsikllarni olib tashlash. Ttsikllarni almashtirish. Ogoxlantiruvchi xabarlar.

R E J A :

1. **ttsikllarni olib tashlash.**
2. **ttsikllarni almashtirish.**
3. **Ogoxlantiruvchi xabarlar.**

**Tayanch so`zlar**: *tsikllarni olib tashlash. tsikllarni almashtirish. Ogoxlantiruvchi xabarlar, Kichik xajmdagi programmalarni ishlab chiqarish usullari. Katta xajmdagi programma loyihalarni tadbiq qilish yo`llari.*

.

tsikllarni olib tashlash (uchirish).

Iloji boricha tsikllardan foydalanmaslik kerak. CHunki ularni ishlatilganda bajarish vaqtini uchdan bir qismiga oshiradi. tsikllarni ikkita va undan ortigini bitta tsiklga joylashtirish orqali ularni sonini kamaytirish kerak. tsikllarni dasturdagi vazifalari yaxshilab tekshirilib chiqilgandan so`ng birlashtirish imkoni bo`ladi.

tsikllarni kamaytirish.

tsikllarni kamaytirish yo`llari bilan ham programmani optimallashtirish mumkin. CHunki tsikllarni bajarishga (parametrni kushishga va tekshirishga) ko`p vaqt sarflanadi. tsikllarni ishlatish programmani bajarilishini vaqtini birdan uchga ko`paytiradi. SHuning uchun hisoblashda tsikllarni kamaytirib programmani tuzish kerak.

tsikllarni kamaytirish yo`llaridan biri 2 va undan yuqori tsikllarni bitta tsiklga keltirishdir. Buning uchun programmalashtirishdan oldin vazifani chuqur taxlail qilish kerak.

tsikllarni bajarilishida ko`proq vaqtni tsiklni ishga tushirish va uning indeksini tekshirishga ketadi. Ichma-ich joylashgan tsikllarni to`g`ri tashkil qilish bilan vaqtini kamaytirish mumkin.

tsikllarni optimalashtirish.

Programmani tezroq bajarishda tsikllarni bajarish vaqti asosiy faktor hisoblanadi. Mag`lumki tsikl ichidagi operatorlar bir necha ming marotaba bajariladi. SHu bajarilishda ozgina samaradorlik ham bir necha mingga ko`payadi.

Ichma-ich joylashgan tsikllrada optimalashtirishni ichki tsikl operatorlaridan boshlash kerak. tsikllar bajarilish vaqtini kamaytirish va ishlatish xotirpsini kamaytirish maqsadida ketma-ket yozilgan bir necha tsikllarini bittaga keltirish ishlatiladi.

Indeksatsiya bilan optimallashtirish.

Indeksatsiyalar bilan ishlashda kompg`yuterni vaqti va uning xotiradagi joyi ko`proq ishlatiladi. SHuning uchun indeksatsiyalarni xamoptimallashtirish programmani optimalashtirishga olib keladi.

Agar bir yoki bir necha operatorlarni ichida indeksli o`zgaruv-
chiga bir necha marta murojat qilinayotgan bo`lsa, u holda bu indeksli o`zgaruvchingi boshqa o`zgaruvchi bilan tenglashtirib olish kerak.
Misol uchun quyidagi ifodani x=(A(I)+1/A(I))+A(I), optimallashtirish uchun quyidagicha yozish mumkin:

AI=A(I) x=(AI+1/AI)+AI

Ogoxlantiruvchi xabarlar.

Ko`pincha kompiliyatsiyani samaradlorligini oshrishda axamiyat bergan holda va dasturni bajarishda ma`lum bir shartlarni ogoxlantirish xabarlarni keltirib

chiqaruvchilar olib tashlanadi. Unga sabab ogoxlanturuvchi xabarlar dasturni boshqarishda xech qanday xalaqit berlmaydi va samaradorlik pasaytirishi sababli ularni hisobga olinmaydi. har safar ko`ramalashni bajarishga bunday xabarlarni chaqirish ularni ancha vaqt sarf bo`ladi. Bu esa dasturni bajarishda va xotira sarfini oshiradi.

Ko`ramalashga ogoxlantiruvchi xabarlarni qo`llamagan holda dastur yaratishni oson usuli bu dasturni samaradorligini oshirgan holda bajarishdir. Ogohlantiruvchi xabarlar ko`pincha to`g`irlashda va qayta qurishda beriladi.

Nazorat savollari:

1. **tsikl**larni qo`llanishini samaradorligi ?
2. tsikllarni qanday optimallashtirish mumkin?
3. tsikllar programmada qanday tartibda joylashishi mumikin?
4. Modulli programmalashtirishni afzallligi nimada?
5. Dasturni loyihalashtirishga tushuncha bering?
6. SHEHM da axborot qanday tartibda kiritiladi?
7. Loyihalashda masalani qo`yilishi qanday loyihalashga ega?
8. Algoritmni tanlash, programmalash tilini tanlash dasturni loyihalashtirishda qanday loyihalashga ega?
9. Nima uchun programmalar qulay o`qiladigan bo`lishlari kerak?
10. Izohlarni tez-tez berilishi shartmi?

9 - Mavzu. Programmaning samaradorligi. Yuklash modullari. Modullar. Kompilyator va mashina xakidagi axborotdan foydalanish.

R E J A :

1. Yuklash modullari. Modullar.

2. Kompilyatorva mashina haqidagi axborotdan foydalanish.

3.Yuklovchi modullar.

**Tayanch so`zlar:** *Yuklash modullari. Modullar. Kompilyator va mashina xakidagi axborotdan foydalanish, Kichik xajmdagi programmalarni ishlab chiqarish usullari. Katta xajmdagi programma loyihalarni tadbiq qilish yo`llari.*

Kompilyatorlash natijasida olinadigan dastur ularni bajarish dasturlash buyumlarida ishlatilishi kerak. Dastur sozlanib testdan o`tkazilgandan so`ng,uni keyinchalik ish bajarish uchun yuklovchi modul yaratish kerak.

Diskdagi yuklovchi modullardan foydalanib aloqa urnatish vaularni kompilyatorlash jarayonini hisobga olmasdan ancha vaqt mebrlashda boshlangich dasturni har bir satrini ko`rib o`tishda kampilyatorlar murakkab jarayon bo`lib,optimallashda unga qo`llanilmaydi.

Modullar.

Modullarni yaratish dasturni bajarish tartibini, uni sozlash, testdan o`tkazish va takomillashtirishni ancha osonlashtiradi. Bu esa dastur yozishni boshidanok dasturlash vaqtidan va mashina vaqtidan yutishga olib keladi. Bu esa dsturchi uchun juda muxim, chunki dasturni boshlangich vaqtidanok mashina vaqtidan dasturning bajarilishi jarayoniga nisbatan ancha tayyorlanadi. Dasturni loyihalash, sozlash, testdan o`tkazishni bir - biriga bog`liq bo`lmagan holda mustaqil bajarish mumkin. Natijada dasturchi mashina vaqtidan yutadi. Dasturdan foydalanishdagi kamchiliklardan biri uning xajmi katta bo`lishiga karatiladi. Agar dasturni chiqarish nuktaga quyib, dasturni biron - bir joyidan buyruq berilsa, u buyruqni samarali bajaradi.

Dasturni ketma-ketlik bilan bajarishda aloqa buyrug`i o`zib qo`yiladi va buning oqibatida xotira va vaqt me`yorlanadi.

Modulni dasturlash - bu model deb ataluvchi mantiqiy qismlarga ajratish jarayonidir.

Bu qismlar alohida ketma - ket loyihalanadi, chunki katta masala bir necha kichik masalalarga bo`linsa uni xal etish yoki yechish ancha yengillashadi.

Agar masalani yuqoridan quyigacha loyihalansa, unda u albatta taxminiy modullar uchun bir necha kichik vazifalarga bo`linadi.

1). Modulg` qo`llanilgan konteksta bog`liq bo`lmagan, holda, to`g`ri va aniq dasturiy modulga erishish kerak.

2). Oldindan modulning ichki ishi to`g`risida xech qanday ma`lumotga ega bo`lmagan holda modul yordamida katta dasturni formalashga intilish lozim.

Tadbiqiy dasturlash va standart jarayonlar omadli modullarga mislodir.

Buni ham yuqoridan pastga qarab bajariladi. Fakt unda vazifalarni boshqarish tilini biladi.

Odatda loyiha "skletini" ni bir kishi bajarish mumkin. Bu esa loyihani butunligini ta`minlaydi va dasturni tushunish oson bo`ladi.

Modulni printsipi qo`llanilaetganligi sababli asosiy dastur qisqa bo`lish va ko`maklashuvchi kichik dasturlarni yaratuvchi va modullovchi kichik va modullarni chiqaruvchi bo`lishi kerak.

Ko`maklashuvchidastur xaqiqiy dastur yaralgunga qadar o`rnini bosuvchi qisqa buyruqlar ketma - ketlikdan iborat.

Ko`maklashuvchi dastur 2 ga bo`linadi.

Soxta va almashinuvchan modullar.

Soxta modul xech qanday ish bajarilmaydi, u faqat chaqiruvchi modullarga boshqaruvni qaytarib beradi.

Vaqtincha almashinuvchi modulni murakkab modul yaratulgunga qadar oddiy qayta ishlashni bajaradi.

Almashinuvchi modul natijani bermagunga qadar chiqaruvchi modulg` ishini davom ettirmasiligi kerak. Ko`maklashuvchi djasturlar dasturning boshqa bo`g`inlarini (segmentga) testdan o`tkazish uchun ham xizmat qiladi.

Bosh dasturni 2 ta muvaffaqiyatli loyihalar ishlab chiqarilgandan so`ng tarkibiy dasturlarning hamma usullari birlashtirilib BDB (bosh dastur birgadasi) tuziladi.

1. Texnik boshqarishda loyihani texnik espexiplarni nazorat qilish.

2. Qo`l ostidagilarni boshqarishda odamlarni hisobotligini va tartibligini nazorat qilish.

3. SHartnoma shartlarini bajarishda buyuruvchi bilan muomolani sozlab turish.

Loyihani qo`llab - quvvatlovchi bibloteka (BPR) sistemaga birlashtirilgan va testdan o`tkazilgan brigada tarkibiy dasturlarini foydalanish uchun hamda saqlash uchun xizmat qiladi.

Ko`p masalalarda tez - tez qo`llanilib turadigan modullar,
standart programmalar bir xil qoida asosida tuziladi (rasmiylashti- riladi), ya`ni ularga murojat qilish, foydalanish va ulardan natija olish bir xil qoida asosida tashkil etiladi. Bu esa programma tuzishda ulardan osonlik bilan foydalanish imkonini beradi.

Modullardan ikki xil usulda: asosiy programmaning zarur bo`lgan qismiga modullarni joylashtirish yo`li bilan: har bir mashinaning o`ziga xos komandari mavjud bo`lib, bu komandalar yordamida modullarga murojat qilish yo`li bilan foydalanish mumkin. Ikkinchi usul ko`proq qo`llaniladi. CHunki bu holda operativ xotirada modullar kutubxona shaklida joylashadi va bu modullardan istilgan vaqtda foydalanish mumkin. SHu sababli modullarni ishdan ozod qilish uch xil adres qo`llaniladi:

1. Absolyut adreslarning qiymatlari modullarning tutgan o`rniga qarab o`zgarmaydi;

2. Ichki adreslar modullarning joylashgan yeriga bog`liq holda
hisoblanadi;

3. Tashqi adreslar boshqa modullarning tutgan o`rniga qarab o`zgaradi;

Absolyut adreslar o`zgarmas ish yachekalari yoki mashina registrlari, ba`zi bir maxsus mashina komandalarining adres qismilaridan iborat bo`ladi. Masalan, surish komandasi hamda komanda shaklida yozilgan o`zgarmas sonlarning adres qismlari.

Ichki adreslar boshqa modullarga o`tish komandalari va standart programmaning o`zida o`zgarmaslar yozilgan joyda uchrashi mumkin.

Tashqi adreslar boshqa modullarga o`tish komandalarida uchraydi.

Mashina xotirasida doim saklanadigan modullar tuplami modullar kutubxonasini tashkil etadi. Kutubxona tarkibiga bir necha unlab modullardan bir necha yuzlab modullargacha kiritish mumkin. Umuman mashinada bajariladigan ishlarning soni programma shaklida yezib yi-ilgan bo`ladi. Bu ishlarning ba`zi birlari mashina zimmasiga yuklansa, mashina qurilmasini murakkablashtirib yuboradi. SHuning uchun kutubxona tuzish maqsadga muvofiqdir.

Kutubxona operativ xotiraga chaqirish va ularni xoxlagan shaklida joylashtirish uchun ikki xil: interpretatsiya va kompilyatsiya usullari mavjud. Kompilyatsiya usuli qo`llanilganda modullarning chaqirish va joylashtirish uchun kompilyatsiya qiluvchi sistema, interpretatsiya usuli qo`llanilganda esa interpretatsiya qiluvchi sistema ishlaydi.

Ikkala usulda ham standart programmalar kutubxonasiga, kutubxona katalogiga va boshqaruvchi programmaga murojat qilinadi.

Interpretatsiya usulning afzalligi shundan iboratki, asosiy yoki modul programmalar xajmi jixatdan katta bo`lganda xotiradan bemalol foydalanish mumkin. Kompilyatsiya usulidan esa barcha modullar va asosiy programma operativ xotiraga joylashgan vaqtda foydalaniladi.

Komplyatsiya usuli bilan ishlaganda kutubxonadan barcha kerakli modullar operativ xotiraga chaqirib, jamlab quyiladi va programma ishlaganda operativ xotiradagi modullardan avtomatik ravishda foydalaniladi. Kam vaqt sarflash bu usulning afzalligi, operativ xotiraning ko`p sarflanishi esa kamchiligi hisoblanadi.

Nazorat savollari:

1. Modullarga tushuncha bering
2. YUklovchi modullar- bu nima?
3. Kompilyator va SHEHM to`g`risida ma`lumotlarni qo`llash?

4. tsikllarni qo`llanishini samaradorligi ?
5. tsikllarni qanday optimallashtirish mumkin?
6. tsikllar programmada qanday tartibda joylashishi mumikin?
7. Modulli programmalashtirishni afzallligi nimada?
8. Dasturni loyihalashtirishga tushuncha bering?
9. SHEHM da axborot qanday tartibda kiritiladi?
10. Loyihalashda masalani qo`yilishi qanday loyihalashga ega?

10 - Mavzu. Programmalarni sozlash. Masalalarni tavsiflash, algoritmni tanlash, taxminiy xatoliklari. Umumiy xatoliklar, xatosiz programmalashtirish. Programmalarni tugirlash. Sintaksiz xatolar.

R E J A:

1. Masalalarni tavsiflash. Algoritmni tanlashdagi xato. Umumiy xatoliklar.

2. Programmalarni to`g`rilash.

3. Sintaksiz xatolar.

**Tayanch so`zlar:** *Masalalarni tavsiflash, algoritmni tanlash, taxminiy xatoliklari. Umumiy xatoliklar, xatosiz programmalashtirish. Programmalarni tugirlash. Sintaksiz xatolar, Kichik xajmdagi programmalarni ishlab chiqarish usullari. Katta xajmdagi programma loyihalarni tadbiq qilish yo`llari.*

Dasturlarni sozlash.

Xatolarni borligini aniqlab ularni to`g`rilash - sozlash deb ataladi. Dasturlarni u yoki bu xatolarda sozlash majburiydir. Aks holda biz uni testdan o`tkazishimiz kerak bo`ladi. Jarayonni sozlash dasturni ishlash usuliga bog`liq, ya`ni foydalaniladigan mashinaga, jarayon sistemasiga, dasturlash tiliga, beriladigan vazifa tarkibiga va xattoki dasturni muayyan xususiyatiga ham boulik bo`ladi. YAna shuni aniq aytish mumkinki, har birxil, qurilma va mashinalar dastur kamchiliklari dasturlash xatolari bilan o`zluksiz bog`liq.

Masalan: Sintaksis xatolar bo`lganda dasturlashni aniq tili orqali oldindan bilib yoki aniqlab olinadi.

Xozirgi vaqtda dasturlarni xajmi katta va murakkab bo`lib bormokda, lekin xatolar o`shaligicha qolmoqda.

Masalalarni tavsiflashdagi xatolar.

Odatda dastur yozib bo`lingandan so`ng foydalanuvchi olinayotgan ma`lumotlarni kerakli ma`lumotlardan fark qilishini bilib oladi.

Dasturga quyiladigan talabni sifatsizligi keyinchalik tayyor dasturni noto`g`ri ko`rsatilgan va berilgan vazifalarni to`g`ri ishlashiga olib keladi.

Biron bir buyuruvchi uchun loyihalashaetgan dasturni uning talablariga mos kelmasligini belgilaridan biri, quyiladigan vazifani tushunmayotganligidandir.

SHuning uchun biron bir dasturni loyihalashdan oldin buyuruvchidan quyiladigan talabni yozma ravishda olishimiz kerak. Bu esa buyuruvchini fikrini bir joyga tuplashga va talabni aniq, ravshan va tushunarli qilib yozib berishiga olib keladi.

Algoritmni tanlashdagi xatolar.

Vazifa oxiri aniq bo`lgandan so`ng, dasturchi unga mos keluvchi algoritm yoki yechish dasturlarini axtaradi. Sifatsiz, noto`g`ri tanlangan algoritm sifatida biz vazifani to`g`ri yechadigan lekin hisoblashga uzoq vaqt sarflaydigan misolni ko`rsatishimiz mumkin.

Algoritmi noto`g`riligini uni sinab kurilgandan so`ng aniqlash mumkin. SHuning uchun dasturni boshqatdan tekshirib chiqish oldini olish uchun algoritmga aloxida loyihalash berish kerak.

Tahlil qilishdagi xatolar.

Bunday xatolar sodir bo`ladigan xatolarni to`liq hisobga olmaganligi va vazifani noto`g`ri yechilishida xosil bo`ladi. Birinchi holatga misol tarikasida katta va kichik kattaliklarni, o`zgaruvchilarni manfiy qiymatda xosil bo`lishiga e`tiborsiz qaralishi natijasida hosil bo`lishidir.

Ikkinchi xolatda odatda yirik va kichik mantiqiy xatolar hisoblanadi. Ulardan:

- O`zgaruvchilarni boshlangich qiymatini vazifasini yo`qligi.
- tsikl yakunini noto`g`ri sharti.
- tsiklni noto`g`ri indeksatsiyalash.
- Initsirlashgan tsiklni shartlarini vazifasini yo`qligi.
- Vazifani yechish jarayonini davom ettirish uchun berilgan algo ritmni shoxlarini noto`g`ri ko`rsatilishi.

Sozlashni tashkillashtirishni zng oson yo`li sozlashdan kam foydalanishga harakat qilish ya`ni xatolarga yo`l qo`ymaslikka erishish.

Umumiy ko`rinishdagi xatolar.

Qancha urinmang baribir tanlangan tilga bog`liq bo`lmagan holda dasturlashda xatolarga yo`l qo`yiladi. Bo`larga quyidagilar kiradi:

-Dasturchi tomonidan mashinani yoki dasturlash tilini bilmasligi sababli bo`ladigan xatolar.

-Algoritmni loyihalashda, bunda dasturda foydalanilgan operatorlar algoritm tomonidan qo`yilgan ketma-ketliklarni noto`g`ri bajarishda kelib chiqadigan xatolar.

-Sintaktik xatolar.

-Sntaktik to`g`ri operatorlarni bajarishda kelib chiqadigan xatolar. YA`ni nolga bo`linada yoki manfiy sondan kvadrat ildiz olishda va noto`g`ri ma`lumot berishda. Barcha ko`rsatilgan xatolar sintaktik xatodan tashqari testdan o`tkazish orqali aniqlanadi. Buning natijasida dastur ustida olib boriladigan ish sozlash bosqichiga o`tadi.

Dasturdagi umumiy xatolar.

N%	Xato turi	Misol
1.	Vazifani noto`g`ri qo`yilishi	Noto`g`ri berilgan vazifani to`g`ri yechish.
2.	Noto`g`ri algoritm	Vazifani noto`g`ri yoki samarasiz echishga olib keluvchi algoritm.
3.	Taxlildagi xato.	Algoritmni noto`g`ri dasturlash.
4.	Semantik xato.	Buyruqni tansiflash tartibini tu-shunmaslik.
5.	Sintaktik xato.	Dasturlash tili yordamida aniqlan-gan qoidani buzish.
6.	Jarayon baxsdagi xatolar.	Hisoblashdagi chegaralovchi shartlarga ko`rsatmalarning yo`qligi.
7.	Malumotdagi xatolar.	Ma`lumotda xosil bo`luvchi o`zgarishlarni noto`g`ri aniqlash.
8.	Hujjatdagi xatolar.	Foydalanuvchining hujjatlari xaqiqiy dasturga mos kelmasligi.

Sodda dasturlash.

Dasturni konikarli sozlashda kodlashni oddiyligi va to`g`riligi katta ta`sir ko`rsatadi. SHuning uchun har xil simvol belgiyu vaxokazolardan foydalanmaslik kerak. CHunki ular dasturni sozlashda katta tuskinlar xosil qiladi. Murakkab dasturning boshlang`ich bosqichlarida oddiy va kichkina bloklar, uni soddalashtirish uchun qo`shib yoziladi. Dasturni qulay o`qishligi sozlashni ham osonlashtiradi.

Dasturni to`g`riligi.

Hamma dasturlar mantiqan olib karaganda ayrim bir ma`lumotlar ta`sir ko`rsatuvchi xududlarni aniq ko`rsatishi, ya`ni dastur ish bajarish kobiliyatiga ega bo`lishi, malumotlarni ko`rsatilgan chegaralarda turganligini aniqlash uchui operatorlarni kiritish imkoniga ega bo`lishi kerak.

Dasturni foydalanishga berishdan oldin uni to`g`riligigi ishonch xosil qilish kerak. Noto`g`rilikni aniqlashni ikkita usuli bor:

1.Dasturni konstruktsiyasi sintaktik xato.

2.Dastur noto`g`ri natijalar ko`rsatmoqda.

Sintaktik xato.

Traslyator yordamida sintaktik xatolarni aniqlash dasturni sozlashda eng kerakli va muxim o`rinlarni egallaydi. chunki qancha ko`p xato bo`lsa shu bosqichda aniqlanib to`g`rilansa,keyinchalik sozlashda va testdan o`tkazishda ish oson bo`ladi. YAna bu xatolarni aniqlashning muximligi shartdir, agar ular aniqlanmasa. keyinchalik dasturni bajarishda katta qiyinchiliklarga olib keladi.

Kompilyator uchun bu xatolar dasturni sinash uchun ko`rib o`tishdan oldin bartaraf etilishi lozim. Berilgan operatorlardagi sintaksis xatolar sifatida quyidagilarni misol qilib ko`rsatishimiz mumkin:

-Tinish belgilarini qo`yishda kerakli belgilarni tashlab ketish.

-Kelishilmagan holda qavslarni qo`yish.

-Kerakli qavslarni tashlab ketish.

-Operatorni to`g`ri shakllantirish.

-O`zgaruvchilar nomini noto`g`ri shakllash.

-Arifmetik operatorlardan noto`g`ri foydalanish.

-Olib qo`yilgan yoki ajratib qo`yilgan so`zlarni noto`g`ri yozish.

Ikkita yoki undan ortiq operatorlarni egallab oluvchi sintaktik xatolarga quyidagilar kiradi:

-Karama qarshi buyruq.

-Tsikl oxirini ko`rsatuvchi shartlar yo`qligi

-Belgilar yo`qligi.

-Massvlarni tavsifini yo`qligi.

-Taqiqlangan o`tishlar.

Agar komplyator ikkita yoki undan ortiq buyruqlarni birma-bir tekshirib chikmasa unda yuqorida ko`rsatilgan xatolar aniqlanmay qolib ketadi. Bir xilgi vaqtda yaxshi sozlovchi komplyator tekshirishga ketadigan vaqtni teng yarmini meyorlab koladi. Maxsus sozlovchi ko`ramalar oddiysiga nisbatan sintaktik xatolarni yaxshi aniqlaydi.

CHunki u o`zaro ta`sir etuvchi buyruqlarni va sintaktik sistemani birma-bir tekshirib chikadi. Sozlovchi ko`ramalar EHM bilan birgalikda keltirilmaydi. Ularni aloxida sotib olish kerak.

Agar sozlash dasturchining ishchi vaqtini 70% ni va mashina vaqtini ko`p qismni egallasa, demak, sozlovchi zarur va u tez orada o`z-o`zigani koplaydi.

Barcha asosiy dasturlash tillarini sozlovchi komplyator bilan ta`minlangan bo`ladi.

Agar dasturda mos keluvchi operatorlar to`g`ri shakllantirilgan bo`lsa, u holda kompilyator ayrim xatolarni aniqlay olmaydi.

Bir qator komplyator dastur ob`ektini yordamchi bloklarini keltirib chiqaradi. (Masalan, diapazonlarni va indekslarni o`zlashtirish tekshiruvchi bloklar). Bu bloklar dasturni bajarishda ayrim tavsifga ega bo`lgan xatolarni ngazorat qiladi. Bunday bloklar qanchalik ko`p bo`lsa xatolarni ugnash jarayoni shuncha oson kechadi.

Nazorat savollari:

1. Nima uchun dasturlar sozlanadi? Dasturni sozlash deganda nimani tushunasiz?
2. Qanday turdagi xatoliklarga yo`l qo`yish mumkin?
3. Dasturdagi umumiy xatoliklarga tushuncha bering?
4. tsikllarni qo`llanishini samaradorligi?
5. tsikllarni qanday optimallashtirish mumkin?
6. tsikllar programmada qanday tartibda joylashishi mumikin?
7. Modulli programmalashtirishni afzallligi nimada?
8. Dasturni loyihalashtirishga tushuncha bering?
9. EHM da axborot qanday tartibda kiritiladi?
10. Loyihalashda masalani qo`yilishi qanday loyihalashga ega?

11 - Mavzu. Programmalarni sozlash. Sozlash turlari. Sozlash vositalari. Umumiy tavsiyalar. Interaktiv xolatda sozlash. Programmalarni tekshirish uchun sozlash modullari. Programmalovchiga maslaxatlar.

R E J A :

1. **Sozlash turlari.**
2. **Sozlash vositalari. Umumiy tavsiyalar.**
3. **Interaktiv xolatda sozlash.**

**Tayanch so`zlar:** *turlari. Sozlash vositalari. Umumiy tavsiyalar. Interaktiv xolatda sozlash. Programmalarni tekshirish uchun sozlash modullari. Programmalovchiga maslaxatlar, Kichik xajmdagi programmalarni ishlab chiqarish usullari. Katta xajmdagi programma loyihalarni tadbiq qilish yo`llari.*

Sozlash turlari.

Sintaktik xatolar haqida xabarlar to`xtashi bilanok sozlash boshlanadi. Sozlash jarayoni boshida oddiy test malumotlaridan foydalanish kerak. Agar bunda to`g`ri natijalar xosil bo`lsa unda juda ko`p murakkab ma`lumotlar bilan ishlab kurish kerak. Agar natijalar noto`g`ri bo`lsa unda quyidagi xollar bo`lishi mumkin:

1. Sintaktik xato yuk, lekin dastur kompilyatsiya bo`lmagan.
2. Dastur ishlayapti, kompilyatsiya bo`lgan, lekin natija chiqar-
   mayapti.
3. Dastur kompilyatsiya bo`lgan ishlayapti, lekin qutilmagan o`zi-
   lishlar bo`lyapti.
4. Dastur kompilyatsiya bo`lgan, ishlayapti, lekin noto`g`ri nati-
   ja chiqaryapti.
5. Dastur tsikl ichida ishlab to`xtab qolmoqda. (zattsiklilasg`)

Quyidagi har bir xolatni ko`rib chiqamiz:

1 - xolat. Bu xolatlarni tuzatishda sistemani yaxshi biladigan xodimlardan masalaxat surashi kerak.

2 - xolat. Bu xolatlar qandaydir logik va sistemali xolatlar orqali bo`lgan bo`lishi mumkin. Masalan, algoritm boshlanishi bilan qiymatiga qarab ketishi mumkin.

Tuzatish yo`llaridan natija xosil qilish segmentini qayta dasturlash usulidir.

3 - xolat. Programma ko`zda tutilgandan oldinroq to`xtab qolishi xolatlardan biridir. Bu xolatlarni topish uchun xatolarni topish usullaridan foydalanish maoqul.

4 - xolat. Bu xolat programma to`g`ri tuzilganligini, lekin unda xatolik borligini ko`rsatadi.

5- Bu xolat qaysi tsikldan to`xtash bo`lmaganligini topish kerak. Buning uchun gumon qilingan tsikldan oldin va keyin chiqarish operatorlarini quyib tekshirish kerak. YUqoridagilarni hisobga olib quyidagi umumiy takliflarni berish mumkin:

- Programma qancha ko`p yozilsa shuncha ko`p xato bo`lishi mumkin.
- tuzatish jarayonini programmani yozish etapida o`ylash kerak.
- Tuzatish jarayonida ishlatilaetgan o`zgaruvchi konstantalar-
  ning ro`yxati bo`lishi kerak.
- Topilgan xatolarni albatta ketma - ket tuzatish kerak.
- har doim chiqarilgan natijalarni yaxshilab ko`rib taxlil ki-
  lish zarur.
- Programma variantlarni sanasi bilan aloxida salanishni taxlil qilish kerak.

Sozlash vositalari.

Dasturni yozish vaqtida uning ichiga kiritiladigan sozlash vosi
talari eng samarali hisoblanadi. Bu xolatda dasturchi tomonidan xatolarni aniqlash tez va aniq bo`ladi.

Dasturlashda quyidagi sozlash vositalari qo`laniladi:

1. Xotira tarkibini taxlil qilish.
2. Algoritm bajarilishini nazorat qilish.
3. O`zgaruvchilardan foydalanishni nazorat qilish.
4. Indekslarni tekshirish.

5. Kichik dasturlardan foydalanish ni nazorat qilish.

6. O`zgaruvchilarni belgilarini tasvirlash.

Xotira tarkibidagilarni yozuvga chiqarish.

Dasturni xozirgi xolatini uni bajarilishiga qadar hisobga olishga yoki belgilashga yordam beradi.

YOzuvdagi ma`lumotlar amaliyotda foydalaniladigan xolatiga umuman mos kelmasligi sababli, sozlash moslamasiga intilish xosil bo`ladi va bu moslama ma`lumotni qulay xolatga (o`qish uchun) keltirib boradi.

Algoritmni bajarish yo`llarini nazorat qilish.

Dasturni bajarilishini mantiqiy yo`lini hisobga oladi. Undan dasturchilar tomonidan berilgan jarayonni bajarilish ketma-ketligini va o`zgaruvchilarni xozirgi vaqtdagi belgisini bajarilishini va hisobga olishini tekshirishda foydalaniladi. Bunday ko`zatishlarning uchta turi mavjud:

1. O`zgaruvchilarni belgilarini nazorat qilish.

2. Kichik dasturlarni chiqarishni hisobga olishni nazorat qilish.

3. Dasturdagi buyruqlarni yetkazishni nazorat qilish.

Ikkinchi turdagi ko`zatish asosan kichik dasturlardan foydalanilgan paytda sozlashda qo`llaniladi.

Har bir muomalada yozuvga kichik dasturni nomi chikadi, undan chiqilganda-asosiy dasturga kaytilganligi haqida ma`lumot chikadi.

Bu turdagi tekshirish dasturchiga dasturiy xatolarni aniqlashdagi barcha kerakli ma`lumot bilan ta`minlab turadi. Uning kamchiligi mashina vaqtini ko`p sarf bo`lishidir, hamda berilaetgan ma`lumotlar bir necha ming qatordan iborat bo`lishidadir. SHuning uchun yuqoridagi kamchiliklardan vokif bo`lish uchun dasturni qisimlarga ajratib belgilangan joylarda uchirib yoqib tekshiriladi.

O`zgaruvchilardan foydalanishda nazorat qilish moslamasi.

Nazorat moslamasi shunday tuziladi-ki, unda o`zgaruvchilar birdaniga emas, balki aniq ro`yxat buyicha ko`rsatilgandek tartib bilan yoziladi.

Indekslarni tekshirish. Massivlarni e`lon qilingan chegaralarni va ularni indekslarni solishtirish orqali nomlangan massivlarni indeksatsiyalanganligini to`g`riligini tekshirish.

O`zgaruvchilarni mazmunini takribiy chiqarish.

Display nomli sozlovchi buyruq orqali bajariladi. CHiqariladigan ma`lumotni to`liq chiqishini ta`minlash uchun o`zgaruvchini nomini va uni xozirgi xolat mazmunini ko`rsatilgan buyruq orqali beriladi.

Interaktiv rejimida sozlash.

Juda ko`p xolatlarda, dasturlar dasturchini mashina bilan o`zaro ta`siri ostida bajariladi, shuning uchun bu xolatlarda qulayliklardan kengroq foydalanish maoqulroq, ya`ni sozlash jarayonini samaradorligini oshirish uchun terminaldan kengroq foydalanish kerak. Paketli sozlash sistemasida qo`llaniladigan usullarni ko`p qismni vaqti taqsimlangan sistemada ham foydalanish mumkin, lekin interaktiv jarayon harakatida qo`shimcha faktorlarni va yangi sozlovchi usullarga yondashishga ham loyihalash berish kerak.

Vaqti taqsimlangan sistemalarda ko`pincha maxsus sozlovchi moslamalardan tashkil topgan bo`ladi. SHuning uchun quyida ko`rsatilgan qulayliklarni o`z ichiga olgan sozlovchi sistema terminaliga ega bo`lish kerak.

1. To`xtatish yoki uchirish kaliti ma`lum sharoitlarda dasturni bajarilishini to`xtatish, uni keyingi xolatini qayd qilish va boshqaruvchi foydalanuvchiga o`zatish.

2. Ruxsat etilmagan jarayonlarni bajarish uchun harakat bo`layotgan xolatlarda foydalanuvchiga boshqaruvchi o`zatish uchun xatolar haqidagi signal yordamida to`xtatib qo`yish.

3. har qanday o`zgaruvchilarni mazmunini tartiblash yoki sozlashni xatolik signali yoki o`z kaliti orqali dastur o`qilganda, unga o`zgartirish kiritish yoki xolatini analiz qilish.

4. Qayta ishga tushirish, ya`ni uzilish sodir etilgan operatordan yoki har qanday operatorlardan boshlab ishga tushirib dastur bajarilishini davom ettirish.

5. Qo`shimcha yoki aloxida qatorlarni o`zgartirish, o`chirish uchun kerak bo`lgan modifitsirlangan dastur: bu harakat sozlash jarayoni davom ettirilishi uchun zudlik bilan ishga tushirilishi kerak.

Nazorat savollari:

1. Qanday sozlash turlarni bilasiz?
2. Sozlash vositalarini vazifasi?
3. Interaktiv xolatda sozlash - bu nima?
4. tsikllarni qo`llanishini samaradorligi?
5. tsikllarni qanday optimallashtirish mumkin?
6. tsikllar programmada qanday tartibda joylashishi mumkin?
7. Modulli programmalashtirishni afzallligi nimada?
8. Dasturni loyihalashtirishga tushuncha bering?
9. EHM da axborot qanday tartibda kiritiladi?
10. Loyihalashda masalani qo`yilishi qanday loyihalashga ega?

12 - Mavzu. Dasturni testdan o`tkazish. Testni loyihalash. Testlash usullari. “Qora va Oq” quti usullari metodologiyasi.

R E J A :

1. **Testni loyihalash.**
2. **Testlash usullari.**
3. **Ok va qora qutilarni metodologiyasi.**

**Tayanch so`zlar:** *Testni loyihalash. Testlash usullari. “Qora va Ok” quti usullari metodologiyasi, Kichik xajmdagi programmalarni ishlab chiqarish usullari. Katta xajmdagi programma loyihalarni tadbiq qilish yo`llari.*

Dasturni sozlash va testdan o`tkazish. Dastur loyihalashda murakkab jarayonlar sifatida sozlash etapi va testdan o`tkazish jarayoni hisoblanadi. Dasturni testdan o`tkazishning maqsadi, ya`ni dasturni sinashdan maqsad, undagi mavjud va yengil xosil bo`lgan xatolarni aniqlashdir. Sozlashdan maqsad xatolarni xosil bo`lish sababini aniqlash va ularni bartaraf etish.

Dasturni sozlash uchun ishni testlash rejasini tuzishdan boshlanadi. Rejani tuzishda xatoni kelib chiqishi va uni harakteriga yondashiladi. Asosiy sabablaridan biri matematik modellarni chuqur ishlab chiqilmaganligi va vazifani bajarish algoritmi to`liq qurilmaganligidir;

Dasturiy blankadagi boshlang`ich ma`lumotlar haqidagi 1-tasavvur kiritish moslamasidagi klaviatura yordamida boshlangich ma`lumotni va dasturni tekshirishda loqaydlilik.

Xatolarni kelib chiqish sabablari turli xilligini hisobga olgan holda ularni 2 ta turga ajratiladi:

1.Sintaktik

2.Semantik

Sintaktik xatolar - dastur tilini tarkibini yozishdagi xatolar (sonlar, o`zgaruvchilar, funktsiyalar, operatorlar, belgi va kichik dasturlar).

Semantik xatolar - kattaliklarni ruhsat etilmagan qiymatla-
ridan foydalanishda va harakatni noto`g`ri tarkibi bilan bog`liq bo`lgan xatolar.

Asosiy dasturlash sistemalarida sintaktik xatolarni aniqlash avtomatlashtirilgan (fortran, paskal, beysik va x.k).

Semantiq xatolarni aniqlash uncha yaxshi yo`lga quyilmagan.

Testlash rejasiga odatda quyidagi tiplar kiradi:

1. Dasturni algoritm sxemasi bilan solishtirish

2. Displey ekranida dasturni vizualg` nazorat qilish yoki das-
turiy blankadagi originalni dastur raspechatkasini o`rganish, solishtirishni vizualg` o`rganish.

3. Dasturni mashina tiliga olib ko`rsatish. Bu etapda sintaktik xatolar aniqlanadi. Dastur listingidagi sintaktik xatolarni fortran, paskalp tilidagi ko`ramalar deagnostik xabar beradi.

Xatolar haqidagi xabarlar har xil EHM larda va sistema versiyalarida (fortran, paskalp, beysik) shakliga qarab har xil bo`lishi mumkin.

Interpretatsiyalovchi beysik - navbatdagi operatorni bajarishda xalaqit qilayotgan ham sintaktik, ham semantiq xatolarni birdaniga diagnostika qiladi.

4. Dasturni kompanovkalash va tashqi aloqalarni taxrirlash.

Tashqi aloqalarni taxrirlash jarayonida tashqi aloqalar taxririy dasturi yoki vazifalarini tartiblovchi shunday sintaktik xatolarni aniqlaydiki, bu xatolarda kichik dasturlarda yozilgan prametrlar qiymati bir biriga mos kelmaydi, aslida yo`q bo`lgan standart dasturlarni chiqaradi. Masalan SIN o`rniga SIH va shunga o`xshash xatolar.

5. Dasturni bajarilishi.

Translyator va tashqi aloqalar taxriri tomonidan barcha xatolar bartaraf etilgandan so`ng keyingi - EHM da dasturni mashina tilida bajarish etapiga o`tiladi: dastur operativ xotiraga yuklatiladi va boshlangich ma`lumotlar kiritilgandan so`ng boshlanadi. Xato borligi haqida belgi paydo bo`lishi sozlash o`tkazishna sabab bo`ladi: belgi yo`qligi dasturda xato yo`qligini bildiradi.

Testlash rejasi boshlangich ma`lumotlarning ruxsat etilgan qiymatlarini xulosasini to`g`riligini ham tekshirishni o`z ichiga jamlaydi.

Testlash metodlari (Usullari).

Odatda dasturni testdan o`tkazishda etaplarga bo`lib o`rganiladi. Bunda har bir modulni tekshirishdan tortib, to butun sistemani yakuniy tekshirishlar kabi etaplarni oladi. Agar bunda biron bir ishonchli ketma - ketlikka yondashilmasa, ishonchli ta`minlovchi dastur olish juda qiyindir. Testlash strategiyasi ikkita usuldan birortasiga asosan bajariladi: odatiy quyidan - yuqoriga qarab testlash, yoki zamonaviy yuqoridan - pastga qarab testlash.

Quyidan - yuqoriga qarab testlash.

Bu usul keng tarqalgan usul bo`lib, unda eng quyi pog`onadagi boshlangich yozilgan modullar tekshiriladi. So`ngra yuqori qatlamdagi elementlar dasturlanadi va testlanadi. Bu jarayon to yozilgan dastur butunlay yakunlanmaguncha davom etadi. Quyidan - yuqoriga qarab testlash usuli xozirgi vaqtda yuqoridan - pastga qarab testlovchi va dasturlovchilar tomonidan qo`lanmayapti. Ularni fikricha bu usulda interfeys va algoritmdagi ko`pgina xatolar aniqlanmay qolib ketmoqda. Bu esa dasturni qayta - qayta o`zgartirishdan so`ng buzishga olib keladi.

Ikkinchi kamchiligi esa: har xil pog`onadagi elementlarni tes-tdan o`tkazishda yangidan yangi testlovchi moslamalarni, drayverlarni va testlovchi ma`lumotlarni talab qilmoqda. Bu esa o`z-o`zigadan dasturlashda katta xajmda mehnat talab qiladi.

Yuqoridan pastga qarab testlash.

Bu testlash usuli yuqoridan pastga qarab dasturlashni, yuqoridan pastga qarab kodlashni qo`shimcha etapi hisoblanadi. Bu usulda oldin asosiy dastur yoziladi va so`ngra past pog`onadagi loyihalanmagan elementlar urin bosuvchi dasturlar bilan almashtiriladi.Bunday skeletli dastur chaqiriluvchi dastur va har qanday malumotlar yo`qligida ham o`z ishini davom ettiradi.Bu tekshirish natijasida bazi xollarda bemani bo`lgan xatolar ham aniqlanadi. Keyingi qadam modul qo`shilishidan iborat bo`lib,unda bu modullar kiruvchi modullarni ko`paytiruvchi bo`lishi ham mumkin,

bu esa kiritish moduli,bazi bir yordamchi modul (oxirgini dasturlash tugash daqiqasiga qadar) bo`lishi mumkin. Bu tekshirishdan so`ng sinash oddiy bir sodda kiruvchi malumotlar bilan o`tkazish mumkin.

Favqulotda xollarda tekshirish.

Dasturni testlashning oxirgi etapi o`zgartirishlarsiz ruxsat etilan chegarasidan tashqarida joylashgan qiymatlaridan iborat bo`lgan malumotlardan foydalangan holda bajariladi. Dasturlarni favqulotda va kutilmagan xollarda tekshirishda bir necha maslaxatlar beriladi.Bu maslaxatlar tekshirish vaqtini tayyorlashga yordam beradi.

Avvalambor taxrirlangan malumotlarni to`g`riligini tekshirish kerak bo`lsa, oldin ruxsat etilgan oblastni ikkala chegarasida yotgan qiymatni oling, chunki shu qiymatlar dasturni ishlashida kiynchiliklar tug`dirish extimoli yuqoriroq. Bir xilgi xolatlarda qiyinchiliklarni bir nechta xatolarni jamlagan malumotlar tug`dirishi mumkin. Xatoning yana bir turi - bu ishlov berilaetgan malumotning birligi yoki keyingi elementidagi xatolardir. SHuning uchun dastur o`z ishini odatiy xolga o`xshab tugatmasligi muxumdir. Buning uchun teslashda yagona bir faylni formalansa yetadi.CHunki bu fayl noto`g`ri elementdan tashkil topadi. Agar bu ishlash testlash bosqichida bajarilgan bo`lsa, das-
turni eksplutatsiya qilishda qiyinchiliklar yuzaga keladi.

" Qora " va "Oq" quti usullari metodologiyasi.

Testdan o`tkazish programmalarini normal xolatda ishlashini garantlovchi jarayondir. Qora quti usuli testdan o`tkazish jarayonida ishlatiladigan usullardan biridir unda programma kibernetik ob`ekt sifatida qabul qilinadi va uning kirish qiymatlari va chiqarish qiymatlari bor deb xolos.

Quti ichida qanday jarayon ketayotganligini hisobga olinmaydi. Quti kirishiga qiymatlar beriladi va chiqish qismida ma`lum natija olishni ko`zda tutiladi.

Ikkinchi usullaridan biri bu oq quti usulidir. Bunda programmani kiritish va chiqarish bo`lgan ob`ekt sifatida qabul qilinadi va kirish qiymatlari qanday qilib chiqarish natijalari aylanish mexanizm (programma matni) ham ma`lum bo`ladi.

Nazorat savollari:

1. Dasturni tesdan o`tkazishning maqsadi
2. Qanday testlash metodlarini (usullarini) bilasiz va ularning vazifalariga harakteristika bering?
3. “Qora” va “Ok” quti usullari metologiyasiga tushushcha bering?
4. Dasturni tesdan o`tkazishning maqsadi
5. Qanday testlash metodlarini (usullarini) bilasiz va ularning vazifalariga harakteristika bering?
6. “Qora” va “Ok” quti usullari metologiyasiga tushushcha bering?
7. tsikllarni qo`llanishini samaradorligi ?
8. tsikllarni qanday optimallashtirish mumkin?
9. tsikllar programmada qanday tartibda joylashishi mumikin?
10. Modulli programmalashtirishni afzallligi nimada?

13 - Mavzu. Programmalarni testlash. Testlash vositalari. Test o`tkazish etaplari. Testlarni rasmiylashtirish.

R E J A :

1. Testlash vositalari.

2. Test o`tkazish etaplari.

3. Testlarni rasmiylashtirish.

**Tayanch so`zlar:** *Testlash vositalari. Test o`tkazish etaplari. Testlarni rasmiylashtirish, testni loyihalash,. testlash usullari. "Qora va Oq" quti usullari metodologiyasi, Kichik xajmdagi programmalarni ishlab chiqarish usullari.*

Dasturni testlash jarayonida muxum shartlardan biri shuki dasturlashga qancha vaqt sarf bo`lgan bo`lsa testlashga ham shuncha vaqt ajratish kerak. CHunki dasturlashda rejalashtirish ishlariga intilish (tendentsiya) mavjud bo`lib, shuning uchun barcha loyxalarni muddatlarini bo`zgan holda bajarilishini tushinish mumkin. Bu esa ishlab chiqishga, kodlashga, sozlashga va dasturni testlashga alohida vaqtlar ajratishni va biron bir jarayonni bajarish vaqtini biror xaftaga surilsa, qolgan etaplarni bajarish mudatlari ham surilishini ko`z oldiga keltirishi kerak.

Test o`tkazish etaplari:

1. Normal sharoitda tekshirish.
2. Ekstremal sharoitda tekshirish.
3. Favqulot xolatlarda tekshirish.

Normal sharoitda tekshirish.

Programmani ishlash sharoitidan chikan holda olingan qiymatlar asosida tekshirish. Programma normal sharoitda to`g`ri qiymatlar chiqarishini ko`rsatish kerak.

Ekstremal sharoitda tekshirish.

Bu tekshirishda programmadagi o`zgaruvchilarning chegaraviy qiy matlari asosida test o`tkaziladi. Masalan: Eng kichik qiymatdan maksimal qiymatga ega

(sonli o`zgaruvchi uchun) hamma belgilarni chop qilishi (belgilik o`zgaruvchilari uchun). Bu tekshirishda o`zgaruvchilarni nolg` qiymati ham bo`ladi.

Favqulot xolatda tekshirish.

Bu tekshirish o`zgaruvchining qiymatlar soxasidan tashqarida yotgan qiymatlar asosida tekshirish qiladi.

Rejalashtirish.

Testdan o`tkazish jarayoni vaqtida programma yozish bilan bir bo`lganligi hisobga olib testdan o`tkazishni ham rejalashtiriladi.

Testlash rejasiga odatda quyidagi tiplar kiradi.

1. Dasturni algoritm sxemasi bilan solishtirish.

2. Displey ekranida dasturni vizualg` nazorat qilish yoki dasturiy blankadagi originalni dastur raspechatkasini o`rganish, solishtirishni vizualg` o`rganish.

3. Dasturni mashina tiliga olib ko`rsatish. Bu etapda sintaktik xatolar aniqlanadi. Dastur listingidagi sintaktik xatolarni fortran, paskalp tilidagi ko`ramalar deagnostik xabar beradi.

Xatolar haqidagi xabarlar har xil EHMlarda va sistema versiyalarida (fortran, paskalp, beysik) shakliga qarab har xil bo`lishi mumkin.

Interpretatsiyalovchi beysik - navbatdagi operatorni bajarishda xalakit kilaetgan ham sintaktik, ham semantiq xatolarni birdaniga diagnostika qiladi.

4. Dasturni kompanovkalash va tashqi aloqalarni taxrirlash.

Tashqi aloqalarni taxrirlash jarayonida tashqi aloqalar taxririy dasturi yoki vazifalarini tartiblovchi shunday sintaktik xatolarni aniqlaydiki, bu xatolarda kichik dasturlarda yozilgan prametrlar qiymati bir biriga mos kelmaydi, aslida yo`q bo`lgan standart dasturlarni chiqaradi. Masalan SIN o`rniga SIH va shunga o`xshash xatolar.

5. Dasturni bajarilishi.

Translyator va tashqi aloqalar taxriri tomonidan barcha xatolar bartaraf etilgandan so`ng keyingi - EMM da dasturni mashina tilida bajarish etapiga o`tiladi: dastur operativ xotiraga yuklatiladi va boshlangich ma`lumotlar kiritilgandan so`ng boshlanadi. Xato borligi haqida belgi paydo bo`lishi sozlash o`tkazishna sabab bo`ladi: belgi yo`qligi dasturda xato yo`qligini bildiradi.

Testlash rejasi boshlangich ma`lumotlarning ruxsat etilgan qiymatlarini xulosasini to`g`riligini ham tekshirishni o`z ichiga jamlaydi. Dasturni testlashning oxirgi etapi o`zgartirishlarsiz ruxsat etilan chegarasidan tashqarida joylashgan qiymatlaridan iborat bo`lgan malumotlardan foydalangan holda bajariladi. Dasturlarni favqulotda va qutilmagan xollarda tekshirishda bir necha maslaxatlar beriladi. Bu maslaxatlar tekshirish vaqtini tayyorlashga yordam beradi.

Avvalambor taxrirlangan malumotlarni to`g`riligini tekshirish kerak bo`lsa,oldin ruxsat etilgan oblastni ikkala chegarasida yetgan qiymatni oling, chunki shu qiymatlar dasturni ishlashida kiynchiliklar tugdirish extimoli yuqoriroq. Bir xilgi xolatlarda qiyinchiliklarni bir nechta xatolarni jamlagan malumotlar tugdirishi mumkin. Xatoning yana bir turi - bu ishlov berilaetgan malumotning birligi yoki keyingi elementidagi xatolardir. SHuning uchun dastur o`z ishini odatiy xolga o`xshab tugatmasligi muxumdir. Buning uchun teslashda yagona bir faylni formalansa yetadi. CHunki bu fayl noto`g`ri elementdan tashkil topadi. Agar bu ishlash testlash bosqichida bajarilgan bo`lsa, dasturni eksplutatsiya qilishda qiyinchiliklar yuzaga keladi.

Dasturni to`g`riligi.

Hamma dasturlar mantiqan olib karaganda ayrim bir ma`lumotlar ta`sir ko`rsatuvchi xududlarni aniq ko`rsatishi, ya`ni dastur ish bajarish kobiliyatiga ega bo`lishi, malumotlarni ko`rsatilgan chegaralarda turganligini aniqlash uchui operatorlarni kiritish imkoniga ega bo`lishi kerak.

Dasturni foydalanishga berishdan oldin uni to`g`riligigi ishonch xosil qilish kerak. Noto`g`rilikni aniqlashni ikkita usuli bor:

1.Dasturni konstruktsiyasi sintaktik xato.

2.Dastur noto`g`ri natijalar ko`rsatmoqda.

Odatda dasturni testdan o`tkazishda etaplarga bo`lib o`rganiladi. Bunda `ar bir modulni tekshirishdan tortib, to butun sistemani yakuniy tekshirishlar kabi etaplarni oladi. Agar bunda biron bir ishonchli ketma - ketlikka yondashilmasa, ishonchli ta`minlovchi dastur olish juda qiyindir. Testlash strategiyasi ikkita usuldan

birortasiga asosan bajariladi: odatiy quyidan - yuqoriga qarab testlash, yoki zamonaviy yuqoridan - pastga qarab testlash. Quyidan - yuqoriga qarab testlash.

Bu usul keng tarqalgan usul bo`lib, unda eng quyi pog`onadagi boshlangich yozilgan modullar tekshiriladi. So`ngra yuqori qatlamdagi elementlar dasturlanadi va testlanadi. Bu jarayon to yozilgan dastur butunlay yakunlanmaguncha davom etadi. Quyidan - yuqoriga qarab testlash usuli xozirgi vaqtda yuqoridan - pastga qarab testlovchi va dasturlovchilar tomonidan qo`lanmayapti. Ularni fikricha bu usulda interfeys va algoritmdagi ko`pgina xatolar aniqlanmay qolib ketmoqda. Bu esa dasturni qayta - qayta o`zgartirishdan so`ng buzishga olib keladi.

Dasturchiga maslaxatlar.

- Nazorat qilish usullarining eng oz mikdordagisidan foydalarning.
- Dasturni sinashda xech qanday kamchiliklarga yo`l kuymagan holda, aqliy ravishda o`tkazing.
- Testlashni iloji boricha ertaroq boshlang.
- Tekshirishda kul mehnatidan foydalaning.
- Sistemani kurilish printsiplarini to`g`riligini tekshirishda oddiy ko`rinishda qo`llashga harakat qiling.
- Testlarni yuqoridan pastga qarab usulidan foydalanishga harakat qiling.
- Navbatdagi har qanday testni yangi ma`lumotlar sinfidan foydalaning.
- Dasturni oddiy, eksteremal va favkulodda xolatlarda tekshiring.
- Sinashga ketadigan vaqtni oldindan rejalashtiring.
- Dasturga kiritilgan o`zgartirishlardan so`ng uni yana testdan o`tkazing.

Nazorat savollari:

1. Testal vositalarini vazifasi
2. Test o`tkazish nechta etapdan iborat?
3. Testlarni rasmiylashtirish deganda nimani tushunasiz?
4. Dasturni tesdan o`tkazishning maqsadi

5. Qanday testlash metodlarini (usullarini) bilasiz va ularning vazifalariga harakteristika bering?
6. “Qora” va “Oq” quti usullari metologiyasiga tushushcha bering?
7. tsikllarni qo`llanishini samaradorligi?
8. tsikllarni qanday optimallashtirish mumkin?
9. tsikllar programmada qanday tartibda joylashishi mumikin?
10. Modulli programmalashtirishni afzallligi nimada?

14 - Mavzu. Programmaning asosiy harakteristikalari. Ishonchlik, samaradorlik, inson faktorini hisobi, tushunishlik.

R E J A :

1. **Ishonchlik.**
2. **Samaradorlik.**
3. **Inson faktorini hisobga olish.**

**Tayanch so`zlar**: *Ishonchlik, samaradorlik, inson faktorini hisobi, tushunishlik, Kichik xajmdagi programmalarni ishlab chiqarish usullari. Katta xajmdagi programma loyihalarni tadbiq qilish yo`llari.*

Ishonchlik.

Programmaning har xil muhitda va qiymatlar bilan ishlash ishonchlik deb ataladi.

Samaradorlik.

Programani oz hajmi bilan ishlash samaradorlik deb ataladi.

Lekin samaralorlik 2-etap bo`lib, birinchi etapda programmani to`g`ri ishlashini tashkil qilish kerak.

Inson faktorini hisobga olish.

Programmani foydalanuvchi ishlatganligi munosabati bilan inson faktorini hisobga olish zarur. SHuning uchun foydalanuvchining faktorlarini ham hisobga olish kerak. Progaramma "do`stona" bo`lishi kerak. Ekranlar uta tez almashishi, ma`lumotlar tez chiqib turishi ham insonga salbiy ta`sir ko`rsatadi. Foydalanuvchi interfeysi (ekraniga) katta e`tibor berish kerak. Ekrandagi ranglar soni ham uncha ko`p bo`lmasligi kerak. Ranglar ham o`z urnida ishlatilishi kerak.

SHriftlar ham uta katta yoki o`ta kichik kichik bo`lmasligi kerak.

Baxolanishi.

Programmani baxolash imkoniyati yaratilishi kerak.

Modifikatsiya kilinishi.

Asosiy harakteristkalaridan biri bo`lib, programmaga o`zgartirishlarni ishlab chiqarish jarayonida amalga oshirishga aytiladi. Programmani qayta yezmasdan qandaydir qismlarini o`zgartiriladi.

Ma`lumotlarni tafsiflash.

Yaxshi tafsiflangan ma`lumotlar dasturni ancha qisqartiradi, demak ma`lumotlar massivini tanlashda zarur bo`lganicha qo`llash kerak. Boshqacha usulda tayanch va ko`rsatkichlardan foydalanish kerak.

Ma`lumotlar haqida kurilaetgan vazifaga mos holda taosurotga ega bo`lish kerak. Hisobda har bir ma`lumot tarkibini hamma tilga o`zgartirish mumkin. Lekin siz muljallagan ma`lumot tarkibini mujassamlagan dasturlash tilidan foydalanishingiz maoqulroq.

Dasturiy foydalanuvchilar ehtiyojlarini qondirish, keng tarqatish va sotish uchun mo`ljallangan.

Xozirgi paytda dasturiy maxsulotlarni ochik (legal) tarqatishning boshqa variantlari ham mavjud, ular yalpi (global) va mintaqaviy kommunikatsiyalardan foydalanish bilan yuzaga keladi.

Freeware-erkin tarkatiladigan foydalanuvchining o`zi qo`llab-quvvatlaydigan bepul dasturlar bularga zarur o`zgartirishlar kiritishga xakli.

Shareware-notijorat (shartli-tulovsiz) dasturlar, ulardan odatda tulovsiz foydalanish mumkin. Bunday maxsulotlardan doimiy foydalanilganda muayyan summa badal tulanadi.

Bir qator ishlab chiqaruvchi OEM - dasturlar (Original Equipment Manufacturer), ya`ni kompyuterlarga o`rnatilgan yoki hisoblash texnikasi bilan birgalikda keltirilgan maxsus dasturlardan foydalaniladi.

Dasturiy maxsulot foydalanishga tegishli ravishda tayyorlanishi zarur texnik hujjatlariga ega bo`lishi, shuningdek davlat ro`yxati kodi mavjud bo`lishi lozim. Faqat shunday sharoitlardagina yaratilgan dasturiy majmua dasturiy maxsulot deb nomlanishi mumkin.

Dasturiy maxsulot-sanoat maxsulotining istalgan turi kabirealizatsiyaga tayyorlangan ommaviy ehtiyojli muayyan muammo (vazifa) ni xal etish uchun o`zaro bog`langan dasturlar majmuasidir.

Dasturiy maxsulotlar quyidagtcha yaratilishi mumkin:

buyurtmaga ko`ra individual ishlanma;

foydalanuvchilar orasida ommaviy tarqatish uchun ishlanma.

Individual ishlanmada firma-ishlab chiqaruvchi muayyan buyurtmachi uchun ma`lumotlarni qayta ishlash o`ziga xosligini hisobga oluvchi dasturiy maxsulotni yaratadi.

Ommaviy tarqatish uchun ishlanmani yaratishda firma ishlab chiqariuvchi, bir tomondan, ma`lumotlarni qayta-ishlashni bajarilaetgan funktsiyalar universalligi, boshqa tomondan, muayyan bir qo`llash sharoitida dasturiy maxsulotning moslanishi va sozlanishini ta`minlash lozim. Dasturiy maxsulotlarning ajralib turuvchi xususiyati uning tizimliligi-jamlikda qo`llanilgan holda amalga oshiriladigan qayta ishlash vazifalarining funktsional tlakonligi va tugalligi bo`lmog`i lozim. Dasturiy maxsulot dasturlashtirishning zamonaviy asbobsozlik vositalari qo`llangan holda loyiha ishlarini bajarish sanoat texnologiyasi asosida ishlab chiqiladi. Uning o`ziga xaosligi axborot va asbobsozlik vositalaridan foydalanishni qayta ishlash xususiyatiga bog`liq holda algoritm va dasturlarni ishlab chiqish jarayonining noebligidadir. Dasturiy maxsulotlarni yaratishga ko`plab mehnat, moddiy, moliyaviy zaxiralar talab etiladi; yuqori malakali mutaxasislar zarur.

Dasturiy maxsulotni tayyorlash (ko`zatish) dasturiy maxsulot ishga layokatligini qo`llab-kuvvatlash, unga yangi versiyalar, o`zgartirishlar kiritish, topilgan xatolarni tugralash va xokazolarni o`z ichiga oladi.

Dasturiy maxsulotlar an`anaviy dasturiy maxsulotdardan farkli ravishda dasturlarni yaratishda beriladigan sifat xususiyatlarining kat`iy belgilangan turkumiga ega emas yoki bu xususiyatlarni oldindan aniq ko`rsatish yoki baxolash mumkin emas,chunki dasturiy vosita ta`minlaydigan bir xil qayta ishlash vazifalari turli ichki ishlanmalarga ega bo`lishi mumkin. Xatto dasturiy maxsulotlarni ishlab

chiqishga sarflanadigan vaqt va `ujjatlarni ham oldindan katta aniqlikda belgilash mumkin emas.

Dasturlarni asosiy tavsiflari quyidagilar;

* algoritmik murakkablik (axborotni qayta ishlash algoritm mantiqi);

* qayta ishlashning amalga oshirilgan vazifalari ishlanmalarning tarkibi va chuqurligi;

* qayta ishlash vazifalarining to`laqonligi va tizimliligi;

* dasturlar fayllari xajmi;

* dasturiy vosita tomonidan qayta ishlashning operayion tizimi va texnik vositalarga talblar;

* diskli xotira xajmi;

* dasturni tushurish uchun operativ xotira o`lchami;

* protsessorlar turlari;

* operatsion tizimlar versiyalari;

* hisoblash tarmoklarining mavjudligi va boshqadar.

Dasturiy maxsulotlarning sifat ko`rsatkichlari xilma-xil, ular quyidagi jixatlarda aks etadi:

* dasturiy maxsulotni qanchalik yazshi (oddiy, ishonchli, samara li)foydalanish mumkinligi;

* dasturiy maxsulotdan qanchalik yengil foydalanish mumkinligi;

* dasturiy maxsulotni qullashda sharoit o`zgarganda undan foydalanish mumkinligi yoki yo`qligi va boshqadar.

Harakatchanlik-dasturiy maxsulotlarda ma`lumotlarni qayta ishlash tizimi texnik majmuasi, opertsion muxit, ma`lumotlarni qayta ishllashning tarmok texnologiyasi predmetli soxa o`ziga xosligi va xokazolardan mustaqillikni anglatadi. Harakatchan (ko`p platformali)dasturiy maxsulot hisoblash tarmogi sharoitida foydalanishga xech qanday cheklanishlarsiz kompyuterlar va operatsion tizimlarning turli modellariga urnatilishi mumkin. Bunday dasturiy maxsulotlarni qayta ishlash funktsiyalari biron-bir o`zgarishsiz ommaviy foydalanish uchun makbuldir.

Ishonchlilik-dasturiy maxsulot ishida uzliksizlik va barqarorlik, qayta ishlashni bajarishning aniqligi, ish jarayonidagi xatolarni oldindan bilish bilan belgilanidi.

Samaradorlik-dasturiy maxsulot faoliyatida uning ham bevosita vazifasi-foydanuvchi talabi, ham foydalanish uchun zarur bo`lgan hisoblash zaxiralari harajatlari nuktai nazaridan baxolanadi.

Inson omilini hisobga olish oxirgi foydalanuvchi uchun do`stona interfeysni ta`minlash,dasturiy vosita tarkibida konteksli bog`liq holda aytib beruvchi yoki ukituvchi tizim, dasturiy vositaga kiritilgan funktsional imkoniyatlarni o`zlashtirish va ulardan foydalanish uchun yaxshi hujjatlar mavjudligini anglatadi.

Nazorat savollari:

1. Programma qanday asosiy harakteristikalarga ega?
2. “Ishonchlik, samaradorlik, inson faktorini xislobi, tushunishlik”-tushunchalarni vazifasini ko`rsating.
3. Dasturni tesdan o`tkazishning maqsadi
4. Qanday testlash metodlarini (usullarini) bilasiz va ularning vazifalariga harakteristika bering?
5. “Qora” va “Oq” quti usullari metologiyasiga tushushcha bering?
6. tsikllarni qo`llanishini samaradorligi ?
7. tsikllarni qanday optimallashtirish mumkin?
8. tsikllar programmada qanday tartibda joylashishi mumikin?
9. Modulli programmalashtirishni afzallligi nimada?
10. Dasturni loyihalashtirishga tushuncha bering?

15 - Mavzu. Programmaning asosiy harakteristikalari. Programmani loyihalashda texnologik jarayoni.

R E J A :

1. **Programma tuzish usullari.**
2. **Programma tuzishning texnologik jarayoni.**

**Tayanch so`zlar:** *Programmani loyihalashda texnologik jarayoni,prosedura,fakt,programma Kichik xajmdagi programmalarni ishlab chiqarish usullari. Katta xajmdagi programma loyihalarni tadbiq qilish yo`llari.*

Programma tuzish usullari va vositalari.

Programmam tuzishninig eng asosiy usullaridan bir bu struktu raliy programma tuzishdir. Bu usulda programma tuzish uchun uchta qism mavjud:

1. Yuqoridan pastga programmam tuzish.
2. Modul programmalashtirish.
3. Strukturali kodlash.

YUqoridan pastga programma tuzishda programmaning yuqori qismdan boshlanadi. Programmaning asosiy qismi tuzilib, quyi qismidagi modullar esa vaqtinchalik fakt nomlari bilan atalgan proseduralar bilan almashtiriladi. Programmani asosiy moduli tuzilib, testidan o`tkazilgandan so`ng ketma-ket vaqtincha yozilga modullarni yozish bilan programma tuzish davom ettiriladi.

Modul prolgrammalashtirishda programmani logik qisimlariga bo`linadi. Bu modullar programmada protseduralar va funktsiyalar orqali amalga oshiriladi.

Strukturali kodlash deganda harbir modulni gorizantal va vertikal qatorlarda to`g`ri nomlanishiga aytiladi. Bu usul yordamida modullardan tuzilgan programmalar ishlaydigan testidan o`tkazishi qulay modifikattsiya qilish uchun qulay programmalar yaratish mumkun.

Programmist vazifani taxlil qilib kerakli bo`lgan algaritmni tanlaydi. Tanlangan algaritm to`liq taxlil qilinadi va uning blok-sxemasi chiziladi.

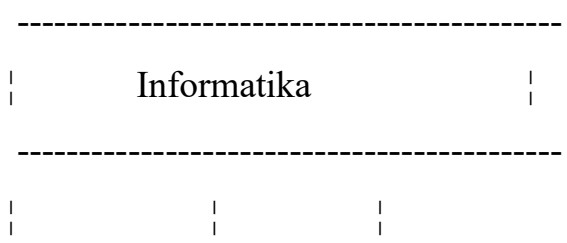
- Programmalashtirish etapi. Dastur yaratish tili tanlanadi.

Programma qabul qilingan algoritimda tuziladi.

- Programmani tuzatish etapi.
- Programmani testidan o`tkazish etapi.

EHMdan foydalanib masalani yechish yaratilgan algoritimga asoslangan holda dastlabki ma`lumotlar ustida avtomatik tarzida amallar bajarib izlardan natija ko`rinishiga keltirish demakdir.

Informatika fani uch tarkibiy qismdan iborat:



Masalani to`g`ri yechib olish uchun zarur bilim va (Algoritm usullar)	EHMda foydalanilgan jami Dasturlar (Dastur)	EHM tarkibiga kirgan, tashqi va chetki qurilmalar (EHM)

Masalani EHMda yechishni bosqichlari.

EHMdan foydalanib "ilmiy- texnik masalani yechish" tushunchasi keng ma`nodagi so`z bo`lib, quyidagi bosqichlarga bo`linadi. Maqsadimiz bosqichlarni qaysi birlarini mutaxassis EHMdan foydalanmasdan va qaysi bosqichlarni EHMdan foydalanib bajarishni aniqlash, hamda bosqichlarni tula o`rganib chiqishdan iborat.

Ilmiy texnik masallarni EHMdan foydalanib yechish bosqichlari:

1. Masalaning qo`yilishi va maqsadining aniqlanishi;
2. Masalani matematik ifodalash;

3. Masalani yechish uslubini ishlab chiqish, sonliy usullarni tanlash;

4. Masalani yechish algoritimini ishlab chiqish;

5. Ma`lumotlarni tayorlash va tarkibini aniqlash (tanlash);

6. Dasturlash;

7. Dastur matnini va ma`lumotlarni axborot tashuvchiga o`tkazish;

8. Dastur xatolarini tuzatish;

9. Dasturni avtomatik tarzda EHMda bojarilishi;

10. Olingan natijalarni izohlash, taxlil qilish va dasturdan foydalanish uchun ko`rsatma yozish;

"Informatika" kursida 1-4 bosqichlar qisqa ma`noda, xususiy xatolar, ko`p uchraydigan murakkab bo`lmagan malumotlar uchun tushutiriladi. Bu bosqichlar tom ma`noda tulaligicha mutaxassislikni egallash davomida maxsus kurslar vositasida urgatiladi.

8 va 9-bosqichlarni bajarishda mutaxassis (EHMdan foydalanuvchi) EHMdan foydalanadi.

7-bosqichda EHMdan foydalanish xam, foydalanmaslik ham mumkin ITMni EHMda yechish bosqichlarini aloxida ko`rib chikamiz.

1-BOSQICH. Masalaning qo`yilishi va maqsadni aniqlanishi. Xalk xujaligining muayan soxasi (texnika, iktisod, lingivistika, ta`lim va x.k) buycha ishlayotgan malakaliy va yetakchi mutaxassis tomonidan bajariladigan ish.

Masalani maqsadni amalga oshirish uchun kerakli ma`lumotlartarkibi (strukturasi), tuzilishi, ifodalanishi aniqlangan bo`lib, ular orasidagi boglanishlar aniq ifodalangan bo`lsa masala qo`yilgan deb aytiladi.

2-BOSQICH. Masalani matematik ifodalash.

Bu bosqichda masalani yechish uchun kerakli va yetarli bo`lgan dastlabki ma`lumotlarni tarkibi, tavsifi, turi, tuzilishi hisobga olingan holda matematik terminlardan ifodalanadi, hamda masalani yechishning matematik terminlarda ifodalanadi, hamda masalani yechimining matematik terminlarda ifodalanadi, hamda

masalani yechishning matematik modeli yaratiladi. Buning uchun har xil matematik aparat ishlatilishi mumkun. Masalani iqtisod soxasidagi mutaxassislar chiziqli dasturlash, dinamik dasturlash, statistik dasturlash, bashorat qilish bilan bog`liq masalalarni yechish mamtematik aparatini bildirish kerak; Texnik soxasidagi mutaxasisliklar oddiy differentsial tenglamalar va ularning tizimlari, mexaniqaning chetki masalalarni, gaz dinamikasiga oid maslalarni, integral ko`rinishidagi masalani ifodalash va yechish uchun ishlatiladigan matematik aparatni to`liq tushunib yotgan bo`lishi kerak.

3-BOSQICH. Masalani yechish usulini ishlab chiqish, sonli usulni tanlash.

4- BOSQICH. Masalani yechish algaritmin yaratish.

5- BOSQICH. Ma`lumotlarni tayyorlash va tarkibini aniqlash.

6- BOSQICH. Dasturlash.

7- BOSQICH. Dastur matnini va ma`lumotlarni axborot tashuvchiga o`tkazish.

8- BOSQICH. Dasturni xatosini tuzatish.

9- BOSQICH. Dasturni avtomatik tarzda EHMda bajarilishi.

10- BOSQICH. Olingan ma`lumotlarni izohlash, taxlil qilish va dasturdan foydalanish uchun yo`riqnoma yozish.

Nazorat savollari:

1. Programmani tayyorlash etaplarini ko`rsating.
2. Programmalashning maqsadini ko`rsating?
3. Unversal programma vazifasini ko`rsating
4. Dasturni tesdan o`tkazishning maqsadi
5. Qanday testlash metodlarini (usullarini) bilasiz va ularning vazifalariga harakteristika bering?

6. “Qora” va “Ok” quti usullari metologiyasiga tushushcha bering?
7. tsikllarni qo`llanishini samaradorligi ?
8. tsikllarni qanday optimallashtirish mumkin?
9. tsikllar programmada qanday tartibda joylashishi mumikin?
10. Modulli programmalashtirishni afzallligi nimada?

16 - Mavzu. Programmaviy ta`minot maxsulot sifatida. Programmaviy maxsulotni loyhalash usullari va komponentlari.

R E J A :

1. Maxsulot kontseptsiyasi.

2. Programmaviy maxsulotni loyhalash usullari.

Tayanch so`zlar: *Programmaviy maxsulotni loyhalash usullari va komponentlari.*

Maxsulot kontseptsiyasi.

Programma maxsulot sifatida xuquq tomonidan himoyalangan tovardir. Programmani xuquq tomonidan O`zbekiston Respublikasining tegishli konunlari himoya qiladi.

Tuzilgan programma maxsulot sifatini olish uchun va uning av-
tori himoyalanishi uchun Respublikada tuzilgan "Algoritm va programmalar fondiga" murojat qilish va undan tuzilgan programmani qayd etish zarur.

Programma maxsulotini tayorlash vositalari va komponentlari.

Programma maxsulotini tayorlashda quyidagilar zarur:

- Taqdimnoma
- Diskda yozilgan bajariluvchi fayl (exe va com fayllarida)
- Programmani matni
- Tushuntirish xati
- Testdan o`tkazish qiymatlari
- Test natijalari
- Modifikatsiya qilish yo`llari

Dasturni va dasturiy komplekslarni loyihalash qisqa vaqt ichida loyihalovchilar yordamisiz foydalaniladigandasturiy maxsulot yaratishni taminlashi lozim. Dasturiy maxsulot sifatida dasturning teksti, mashinada ifodalovchi sifati ko`rinishi (magnit disk) va dasturni hujjatlari sifatida kuriladi. Dasturiy maxsulotlar dastur va algoritm

fondida hisobga olinadi. Ularning vazifasiga foydalanuvchi uchun dastur va uning hujjatlarini nusxalarini chiqarib berish kerak.

Dasturiy maxsulot sifatiga quyidagilar talab qilinadi: Funktsionalligi, ishonchligi, foydalanishga qulayligi (kerakli hujjatlarni yaratishda va dasturiy kompleksni loyihalashni to`g`ri bajarish hisobiga).

Murakkab dastur va dasturiy komplekslar uchun dastur loyihalash algoritm va ma`lumotlar tarkibini loyihalash bilan birgalikda bajariladi.

Loyihalashda vazmfani bir nechta kichik vazifalarga bo`lib, ularni yechish uchun dasturiy modul xosil qilinadi. Bu xolatda dastur ko`p moduli struktura sifatida loyihalanadi. Ko`p modulli dasturlarni ikkita loyihalash usuli mavjud: kutariluvchi va pasayuvchi. Kutariluvchi loyihalash (pastdan yuqoriga qarab loyihalash). Bunda bir nechta yirik modullarga ajratib ularni ayrim funktsiyalari umumiy dasturda amalga oshmriladi.

Modullarni ajratilaetgan vaqtda foydalaniladigan funktsiyalarni tushinish uchun osonligicha, ma`lumotlar tarkibini oddiyligicha, berilgan funktsiyalarni amalga oshirish uchun modul va dasturlarni tayyor xolatda borligicha, yangi maksdlarda tayyor dasturni o`zgartirishga imkoniyat borligicha va kelajakdagi modulning o`lchami va loyihalashiga yendashadi.Kutariluvchi loyihalashda har bir modul avtonom ravishda dasturlanadi, testdan o`tkaziladi va sozlanadi. SHundan keyin aloxida bo`lgan modullar podsistemaga boshqaruvchi modullar yordamida birlashtiriladi. Bu xolatda modullar chaqirish ketma ketligi, kiritish, chiqarish, natija va ma`lumotlar nazorati yordamida birlashtiriladi. SHunga asosan o`z navbatida podsistemalar murakkab sistema va umumiy dasturiy kompleksga birlashtiriladi. Bunda modullararo aloqalar to`g`riligiga kommpleks sozlashdan o`tkaziladi.

Bunday yendashishlar uncha murakkab bo`lmagan dasturlarni loyihalashda foydalanish maslaxat beriladi.Agar kichik dasturlarning o`lchamlari katta bo`lmasa, unda kichik dasturlarni ajratib ularni loyihalashni boshlagan maoqulokdir.

Kutariluvchi loyihalashning asosiy nuqsonlari sifatida modullarni yagona sistemaga birlashtirishda, loyihalashning boshida yo`l qo`yilgan xatolarni aniqlash va ularni bartaraf etishni kurish mumkin. Undan tashqari aloxida bo`lgan modullarni

butun sistemani tarkibini tasavur etmagan holda yaratish mumkin. Bu esa birlashtirishni murakkablashtiradi. Juda murakkab bo`lgan dasturlarni loyihalashda pasayivchi loyihalash maslaxat beriladi. Bu usulda yechilayotgan masalani umumlashtirish pog`onalariga bo`ysunishiga asoslanadi. Umumlashtirish pog`onalarini bo`ysunish sxemasi dasturini fikrini oldin nima qilish kerak va undan so`ng qanday qilish kerakligiga to`g`rilaydi. Asosiy dastur yuqori pog`onadagi dastur, chaqiriklarni boshqaruvchi ancha quyi pog`onaga yoziladi. Bu quyi pog`onadagi modullar o`z navbvtida yana ham quyi pog`onada bo`lgan modullar chaqiriklarni boshqaradi. Bunday loyihalash maydoni murakkab va nozik bo`lgan dasturlarni uncha katta bo`lmagan oddiy modullardan yaratishga yordam beradi: vazifa o`lchami umumlashtiruvchi modullar sonida tasvirlanadi.

Dasturni yuqori sifatiga birinchi urinda loyihalash vaqtida algoritm sxemasini chuqur ishlab chiqish hisobiga erishiladi. Bu avvalambor dasturni bexatoligi, dasturchini xatolar yo`qligiga ishonch va foydalanuvchini dasturni to`g`rililigiga ishonchidir. Dasturni bu xatoligiga ishonchlilik, uni tushunarligi, odiyligi, oson o`qilishi, mualiflar va foydalanuvchilar tomonidan oson interpretatsiyalashdir. CHunki xatolar uni yaratishda va foydalanishda ham paydo bo`lishi mumkin.

Loyihalash vaqtida xato kilmaslikni iloji yuk, shuning uchun ularni oldini olish uchun quyidagi maslaxatlar beramiz. Bu maslaxatlar yordamida siz tezda xatolarni va ularni sabablarini aniqlashingiz, hamda oldini olishingiz mumkin.

1. Qo`yilgan vazifani, uni matematik modelini yaratish vaqtidayok chuqur, to`liq va yaxshi tushunishga intilish kerak. Bunday intilishlar dastur ob`ektlari o`rtasidagi mantiqiy o`zaro tasirlarni tushunishga olib keladi.

2. Vazifalarni yechish algoritmi EHM da hisoblash jarayonlarini kobiliyatini to`liq hisobga olgan holda ishlab chiqiladi.

3. Algoritmni loyihalash vaqtida iloji boricha tushunarli va sodda bo`lishiga intilish kerak. Bu dasturni yozishdagi dasturlash tili shakliga ham bogligdir.Tarkibiy dasturlashda qo`llaniladigan standart usullar, dasturni tushunarli qiladi, ammo bir xilgi xollarda u nozik va samarasiz bo`lib kolish xollari ham uchraydi. SHuning uchun sodda va tushunarli dastur samarali dasturga nisbatan ustun turadi. Tarkibiy

dasturlash sistemali yendashishning asosiy printsplariga asoslanadi: dastur mayda qadamlardan tashkil topishi kerak. Qadam o`lchamlari dasturchi tomonidan shu qadamda qo`llaktgan yechimlar soniga asosan ulchanadi. Murakkab vazifalar yetarli ravishda sdda, qabul qilishga oson bo`lgan blokka bo`linadi. Ularning har biri bittadan kirish va chiqishga ega bo`ladi.Dasturning mantiqiy oddiy tarkibiy boshqaruvchi bazalarning eng oz soniga tayanadi. Tarkibiy dastur bittadan kirish va chiqishdan iborat bo`lgan ketma-ket yoki bir-biriga kiydirilgan bloklar tizimidan tashkil topgan. Bunda bloklar o`lchami dasturlash tili (operator) ning elementar tallifi bugimigacha yetadi. Eng sodda dasturga ega bo`lish uchun, siz loyihalash vaqtida ma`lumotlar tarkibini aniqlab olishingiz kerak. CHunki bu tarkiblar ustidan ishlagandan so`ng siz intilgan natijaga erishasiz. Dastur ma`lumotlar bilan algoritmlar o`rtasidagi teskari tasvir sifatida ko`rinishi kerak. Dasturda kiritish chiqarish jarayonlari, tashkil etishga, dastur tashqarisidan kiritilaetgan boshlangich ma`lumotlarni raspechatkasi va yozishda ma`lumotlar tarkibini va ularni interpretatsiyasini tablitsa6 grafika va x.k sifatida ko`rsatish kerak. O`zgaruvchilarni initsializatsiyalash ni ulardan foydalanishdan oldin bajarish lozim. Ma`lumotlarni to`g`riligini, ularni algoritmga kiritishda tekshirish dasturni sifatini oshiradi.O`zgaruvchilarni solishtiruvchilari ma`noli yuklashlardan iborat bo`lishi kerak. Juda ko`p xollarda o`zgaruvchilarni tasvirlovchilar dasturda oson rasshifrovkalovchi so`zlarning bosh harflaridan tashkil topgan bo`ladi.

Boshlangich ma`lumotlarni tubanlashishida dasturni to`g`ri bajarilishiga intilish kerak.Bunda saralovchi dastur bitti elementdan tashkil topgan ro`yxatni saralashi kerak; matritsalar bilan ishlovchi dastur matritsa bitta elementdan tashkil topganda qayta shakllanishi kerak; noto`g`ri boshlangich ma`lumotga ega bo`lgan dastur xatolar borligi haqida xabar beruvchi vazifa bilan tugashi kerak. Dasturni o`qish oson kechishi uchun aloxida so`zlar va jumlalar urtasida joy(probel) qoldirish maslaxat beriladi.

Dasturni tarkibiy qismida dastur to`g`rismda ma`lumot beruvchi ilovalar bo`lishi kerak.

Nazorat savollari:

1. Programma maxsulotini tayyorlash vositalari va komponentlariga tushuincha bering.
2. Programmamiviy maxsulotni loyihalash usullarini vazifasini ko`rsating.
3. Dasturni tesdan o`tkazishning maqsadi
4. Qanday testlash metodlarini (usullarini) bilasiz va ularning vazifalariga harakteristika bering?
5. “Qora” va “Oq” quti usullari metologiyasiga tushushcha bering?
6. tsikllarni qo`llanishini samaradorligi?
7. tsikllarni qanday optimallashtirish mumkin?
8. tsikllar programmada qanday tartibda joylashishi mumikin?
9. Modulli programmalashtirishni afzallligi nimada?
10. Dasturni loyihalashtirishga tushuncha bering?

17 - Mavzu. Programmaviy ta`minot maxsulot sifatida. Programmaviy maxsulotni loyihalash va analiz qilish usullari.

R E J A :

1. **Programmaviy maxsulotlarni konstruktsiyalash.**
2. **Programmaviy maxsulotlarni baxolash.**
3. **Ishlatilishi.**

**Tayanch so`zlar:** *Programmaviy maxsulotni loyihalash va analiz qilish usullari. Kichik xajmdagi programmalarni ishlab chiqarish usullari. Katta xajmdagi programma loyihalarni tadbiq qilish yo`llari.*

Foydalanish xususiyati va foydalanuvchilar kategoriyalariga ko`ra barcha dasturlarni ikki sinfga - utilitar dasturlar va dasturiy maxsulotlarga bo`lish mumkin.

Utilitar dasturlar - shu dasturlarni ishlab chiqaruvchilar extiyojini qondirish uchun mo`ljallangan. Ko`proq utilitar dasturlar ma`lumotlarni qayta ishlash texnologiyasida servis rolini bajaradi yoki keng tarkalish uchun mo`ljallanmagan funktsional vazifalarni xal etish dasturlari bo`ladi.

Dasturiy maxsulotlar foydalanuvchilar ehtiyojlarini qondirish, keng tarqatish va sotish uchun mo`ljallangan.

Xozirgi paytda dasturiy maxsulotlarni ochik (legal) tarqatishning boshqa variantlari ham mavjud, ular yalpi (global) va mintaqaviy kommunikatsiyalardan foydalanish bilan yuzaga keladi.

Freeware-erkin tarkatiladigan foydalanuvchining o`zi ko`llab-kuvvatlaydigan bepul dasturlar, bo`larga zarur o`zgartirishlar kiritishga xakli.

Shareware-notijorat (shartli-tulovsiz) dasturlar, ulardan odatdatulovsiz foydalanish mumkin. Bunday maxsulotlardan doimiy foydalanilganda muayyan summa badal tulanadi.

Bir qator ishlab chiqaruvchilar (OEM-dasturlar Original Equipment Manufacturer), ya`ni kompyuterlarga o`rnatilgan hisoblash texnikasi bilan birgalikda keltirilgan maxsus dasturlardan foydalaniladi.

Dasturiy maxsulot foydalanishga tegishli ravishda tayyorlanishi zarur texnik hujjatlariga ega bo`lishi, shuningdek davlat ro`yxati kodi mavjud bo`lishi lozim. Faqat shunday sharoitlardagina yaratilgan dasturiy majmua dasturiy maxsulot deb nomlanishi mumkin.

Dasturiy maxsulot-sanoat maxsulotining istalgan turi kabi realizatsiyaga tayyorlangan ommaviy ehtiyojli muayyan muammo (vazifa) ni xal etish uchun o`zaro bog`langan dasturlar majmuasidir.

Dasturiy maxsulotlar quyidagtcha yaratilishi mumkin:

buyurtmaga ko`ra individual ishlanma;

foydalanuvchilar orasida ommaviy tarqatish uchun ishlanma.

Individual ishlanmada firma-ishlab chiqaruvchi muayyan buyurtmachi uchun ma`lumotlarni qayta ishlash o`ziga xosligini hisobga oluvchi dasturiy maxsulotni yaratadi.

Ommaviy tarqatish uchunishlanmani yaratishda firma-ishlab chiqariuvchi, bir tomondan, ma`lumotlarni qayta ishlashni bajarilaetgan funktsiyalar universalligi, boshqa tomondan, muayyan bir qo`llash sharoitida dasturiy maxsulotning moslanishi va sozlanishini ta`minlash lozim. Dasturiy maxsulotlarning ajralib turuvchi xususiyati uning tizimliligi-jamlikda qo`llanilgan holda amalga oshiriladigan qayta ishlash vazifalarining funktsional tlakonligi va tugalligi bo`lmogi
lozim.

Dasturiy maxsulot dasturlashtirishning zamonaviy asbobsozlik vositalari qo`llangan holda loyiha ishlarini bajarish sanoat texnologiyasi asosida ishlab chiqiladi. Uning o`ziga xosligi axborot va asbobsozlik vositalaridan foydalanishni qayta ishlash xususiyatiga bog`liq holda algoritm va dasturlarni ishlab chiqish jarayonining noyobligidadir. Dasturiy maxsulotlarni yaratishga ko`plab mehnat, moddiy, moliyaviy zaxiralar talab etiladi; yuqori malakali mutaxasislar zarur.

Dasturiy maxsulotni tayyorlash (ko`zatish) dasturiy maxsulot ishga layokatligini qo`llab-kuvvatlash, unga yangi versiyalar, o`zgartirishlar kiritish, topilgan xatolarni tugralash va xokazolarni o`z ichiga oladi.

Dasturiy maxsulotlar an`anaviy dasturiy maxsulotdardan farkli ravishda dasturlarni yaratishda beriladigan sifat xususiyatlarining kat`iy belgilangan turkumiga ega emas yoki bu xususiyatlarni oldindan aniq ko`rsatish yoki baxolash mumkin emas, chunki dasturiy vosita ta`minlaydigan bir xil qayta ishlash vazifalari turli ichki ishlanmalarga ega bo`lishi mumkin. Xatto dasturiy maxsulotlarni ishlab chiqishga sarflanadigan vaqt va hujjatlarni ham oldindan katta aniqlikda belgilash mumkin emas.

Dasturlarni asosiy tavsiflariquyidagilar;

* algoritmik murakkablik(axborotni qayta ishlash algoritm mantiqi);
* qayta ishlashning amalga oshirilgan vazifalari ishlanmalarning tarkibi va chuqurligi;
* qayta ishlash vazifalarining to`laqonligi va tizimliligi;
* dasturlar fayllari xajmi;
* dasturiy vosita tomonidan qayta ishlashning operayion tizimi va texnik vositalarga talblar;
* diskli xotira xajmi;
* dasturni tushurish uchun operativ xotira o`lchami;
* protsessorlar turlari;
* operatsion tizimlar versiyalari;
* hisoblash tarmoklarining mavjudligi va boshqadar.

dasturiy maxsulotlarning sifat ko`rsatkichlari xilma-xil, ular quyidagi jixatlarda aks etadi:

* dasturiy maxsulotni qanchalik yaxshi (oddiy, ishonchli, samarali) foydalanish mumkinligi;
* dasturiy maxsulotdan qanchalik yengil foydalanish mumkinligi;
* dasturiy maxsulotni qullashda sharoit o`zgarganda undan foydalanish mumkinligi yoki yo`qligi va boshqadar.

Harakatchanlik-dasturiy maxsulotlarda ma`lumotlarni qayta ishlash tizimi texnik majmuasi, opertsion muxit, ma`lumotlarni qayta ishllashning tarmok texnologiyasi predmetli soxa o`ziga xosligi va xokazolardan mustaqillikni anglatadi. Harakatchan (ko`p platformali) dasturiy maxsulot hisoblash tarmogi sharoitida foydalanishga xech qanday cheklanishlarsiz kompyuterlar va operatsion tizimlarning turli modellariga urnatilishi mumkin. Bunday dasturiy maxsulotlarni qayta ishlash funktsiyalari biron-bir o`zgarishsiz ommaviy foydalanish uchun makbo`ldir.

Ishonchlilik-dasturiy maxsulot ishida o`zliksizlik va barqarorlik, qayta ishlashni bajarishning aniqligi, ish jarayonidagi xatolarni oldindan bilish bilan belgilanidi.

Samaradorlik-dasturiy maxsulot faoliyatida uning ham bevosita vazifasi-foydanuvchi talabi, ham foydalanish uchun zarur bo`lgan hisoblash zaxiralari harajatlari nuktai nazaridan baxolanadi.

Inson omilini hisobga olish oxirgi foydalanuvchi uchun do`stona interfeysni ta`minlash,dasturiy vosita tarkibida konteksli bog`liq holda aytib beruvchi yoki ukituvchi tizim, dasturiy vositaga kiritilgan funktsional imkoniyatlarni o`zlashtirish va ulardan foydalanish uchun yaxshi hujjatlar mavjudligini anglatadi.

Modifikatsiyalanganlik-dasturiy maxsulot ishida o`zgartirishlar kiritish, masalan, qayta ishlash funktsiyalarini kenggaytirish, qayta ishlashning boshqa texnik bazasiga o`tish va boshqadarga kobillikni anglatadi.

Kommunikativlik-dasturiy maxsulot ishida boshqa dasturlar bilan eng ko`p integrayiyasi, umumiy takdim etish shakllarda ma`lumot almashuvini ta`minlash (ma`lumotlar bazasining eksport va importi, qayta ishlash ob`ektlarini tadbiq etish yoki boglash va boshqadarga asoslangan).

Dasturiy maxsulotlar bozori mavjudligi sharoitlarida muxim xususiyatlar quyidagilardir:

* narx;

* sotilgan maxsulot soni;

* bozorda turish vaqti(savdo davomiyligi);

* firma-ishlab chiqaruvchi va dasturlari mashxurligi;

* xuddi shunday dasturiy maxsulotlarning mavjudligi.

Ommaviy tarkatiladigan dasturiy maxsulotlar bozor ehtiyoji va konstrukturasini (raqiblar dasturlari mavjudligi va narxlari) `isobga oluvchi narxlarda sotiladi. Firma olib boradigan marketing ishlari katta loyihalashga ega, u quyidagilarni o`z ichiga oladi:

* bozorni egallash uchun narx-navo siesatini shakllantirish;

* dasturiy maxsulotni keng reklama qilish;

* dasturiy maxsulotni sotish uchun saovdo tarmoklarini barpo qilish (dilerlik va distribyuterlik markazlari deb ataladi);

* dasturiy maxsulot foydalanuvchilarga kafolatli xizmat ko`rsatishni ta`minlash, tezkor liniyani yaratish (dasturiy maxsulotlardan foydalanish jarayonida yuzaga kelgan savollarga tezkor javob);

* dasturiy maxsulot foydalanuvchilarini o`qitish.

Nazorat savollari:

1. Foydalanish xususiyati va foydalanuvchilar kategoriyaga ko`ra barcha dasturlar necha sinfga bo`linadi?
2. Dasturiy maxsulot bu nima?
3. Dasturlarni asosiy tavsiflarini ko`rsating.
4. Dasturni tesdan o`tkazishning maqsadi
5. Qanday testlash metodlarini (usullarini) bilasiz va ularning vazifalariga harakteristika bering?
6. “Qora” va “Ok” quti usullari metologiyasiga tushushcha bering?
7. tsikllarni qo`llanishini samaradorligi ?
8. tsikllarni qanday optimallashtirish mumkin?
9. tsikllar programmada qanday tartibda joylashishi mumikin?
10. Modulli programmalashtirishni afzallligi nimada?

MUNDARIJA