



DASTURLASH TILLARINING RIVOJLANISH TARIXI

Yuldasheva Dilnoza Bekmurodovna
filologiya fanlari bo'yicha falsafa
doktori (PhD), dotsent,

Mamatqulov Daler
"Iqtisodiyot" fakulteti, 1 kurs talabasi
Samarqand iqtisodiyot va servis
instituti

***Annotatsiya:** Ushbu maqolada dasturlash tillarining rivojlanish tarixini o'rganishning ahamiyati haqida bo'lib, zamonaviy dasturlash tillarini o'rganish bo'yicha tasniflanadi.*

***Kalit so'zlar:** Punch kartalari, dasturiy vosita, kodlar, protsessor, assembler tili, FORTRAN.,*

Markaziy protsessor tomonidan bajariladigan ikkilik ko'rsatmalar to'plamini o'rnatish uchun, kompyuter davrining boshida, XX asrning 40-yillarida, dasturchilardan turli xil kalitlardan jismoniy foydalanish va turli xil o'tish moslamalarining joylashishini o'zgartirish talab qilingan. Bu juda noqulay edi, chunki kompyuter dasturini kompyuterga qo'lda kiritish kerak edi.

167 kvadrat metr maydonni egallagan 18000 ta vakuum naychalari va diodlardan iborat 27 tonnalik ENIAC dunyodagi birinchi haqiqiy kompyuter hisoblanadi. Mashina sekundiga 5000 operatsiyani bajarishi mumkin edi.

50-yillarda kompyuterlar Punch kartalaridan foydalanishni boshladilar. Punch kartalari karton kartalar bo'lib, kerakli ko'rsatmalarni o'rnatish uchun kerakli joylarda teshiklar qilish kerak edi.

Past darajadagi tillar.

Kompyuter davrining boshida kompyuterlarni dasturlash uchun mashina kodi ishlatilgan. Bu protsessor ko'rsatmalarining to'g'ridan-to'g'ri to'plami, shuningdek, u yoki bu ko'rsatma uchun dalillar edi. "Salom, dunyo!" mashina kodida.

Mashina kodlarida yozish uchun siz kompyuterning ichki tuzilishini va Markaziy protsessorning tafsilotlarini yaxshilab bilishingiz kerak. Shuningdek, turli kompaniyalarning turli xil Markaziy protsessorlari o'zlarining buyruqlar to'plamiga ega edilar, bu esa dastur Markaziy protsessorning o'ziga xos modeli uchun mashina kodlarida yozilishiga olib keldi.

Assembler tili

Assembler tilining tug'ilgan yili 1949 yil deb hisoblanishi mumkin. Assembler tili-bu dasturlarni mashina kodida yozishni osonlashtiradigan yozuv



tizimi. Buyruq kodlari o'rniga "MOV" yoki "ADD" kabi maxsus belgilar (mnemonika) ishlatiladi. Shuningdek, turli xil sanoq tizimlaridan foydalanish, shuningdek belgilarga ramziy nomlar berish mumkin edi. Ekranda " Salom, dunyo!" (NASM Linux dialektlari).

```
SECTION .data
msg db "Hello, world!",0xa
len equ $ - msg
SECTION .text
global _start
_start: ; Dasturga kirish nuqtasi
mov eax, 4 ; 'write' tizim qo'ng'irog'i
mov ebx, 1
mov ecx, msg ; Ma'lumotlarga ko'rsatgich
mov edx, len ; Ma'lumotlar soni
int 0x80 ; Yadro chaqiruvi
mov eax, 1 ; '_exit' tizim qo'ng'irog'i
mov ebx, 0 ; Qaytish 0 (hammasi yaxshi)
int 0x80 ; Yadro chaqiruvi
```

Shuni ta'kidlash kerakki, assembler tilidagi ko'rsatmalar birma-bir mashina kodlaridagi ko'rsatmalarga tarjima qilinadi (yoki hech bo'lmaganda dastlab shunday bo'lgan). Assembler tili protsessor uchun ko'rsatmalarni yozib olishning yanada qulay shaklini taqdim etadi.

FORTTRAN va tuzilmagan dasturlash tillari

Vaqt o'tishi bilan kompyuterlar yanada samarali va arzonlashdi. Kompyuterlar nafaqat harbiy va olimlarga, balki turli xil tijorat kompaniyalariga ham qiziqish uyg'otdi, nafaqat ilmiy hisob-kitoblar bilan shug'ullanadigan tobora murakkab dasturlarni ishlab chiqish zarurati tug'ildi. Mavjud tillar (assembler va mashina buyruqlari tili) katta va murakkab dasturlarni yozishga imkon bermadi, bundan tashqari, biznes uchun dasturlarni yozish narxini pasaytirish masalasi paydo bo'ldi.

Bularning barchasi kompyuter fanini rivojlantirish zarurligini va murakkab dasturlarni yozishga va ularni ishlab chiqarish, amalga oshirish va qo'llab-quvvatlashga kamroq vaqt va pul sarflashga imkon beradigan mutlaqo yangi dasturlash tillarining paydo bo'lishini shakllantirdi.

Admiral Greys Xopper, yuqori darajadagi dasturlash tillaridan birini ishlab chiquvchi, kompyuter dasturlarini ishlab chiqish uchun yangi vositalarga ehtiyoj



borligini shunday ta'riflagan:

“Men bir paytlar matematika professori bo'lganman. O'sha paytda men matematikani o'rgana olmaydigan talabalar borligini aniqladim. Keyin menga tadbirkorlarga kompyuterlarimizdan foydalanishni osonlashtirish vazifasi topshirildi. Ma'lum bo'lishicha, savol ular matematikani o'rgana oladimi yoki yo'qmi emas, balki ular xohlaydimi yoki yo'qmi. Ularning ko'plari: "ushbu belgilarni tashlang — ular nimani anglatishini bilmayman, ularni o'rganishga vaqtim yo'q" va ma'lumotlarni qayta ishlaydigan odamlar matematik yozuvlardan foydalanishlari kerakligini e'lon qilganlar uchun men maslahat beraman. boshlash uchun vitse-prezident yoki polkovnik yoki admiralning matematik yozuvlarini o'rgatish. Sizni ishontirib aytamanki, men allaqachon sinab ko'rdim.”

1953 yil oxirida IBM xodimi jon Backus FORTRAN deb nomlangan IBM 704 kompyuterini dasturlash uchun assembler tiliga yanada amaliy alternativani taklif qildi. FORTRAN tili uchun birinchi kompilyator 1957 yil aprel oyida ishlab chiqilgan.

FORTRAN tili juda mashhur yuqori darajadagi tilga aylandi va kompyuter dasturlarini ishlab chiqish, amalga oshirish va qo'llab-quvvatlash samaradorligini sezilarli darajada oshirdi.

Yuqori darajadagi tillar kontseptsiyasi dasturiy ta'minotni ishlab chiqish jarayonini sezilarli darajada soddalashtiradi va tezlashtiradi. Yuqori darajadagi tillarning asosiy xususiyati mavhumlikdir, ya'ni bunday ma'lumotlar tuzilmalari va ulardagi operatsiyalarni qisqacha tavsiflovchi semantik konstruktsiyalarni kiritish, ularning tavsiflari mashina kodida (yoki boshqa past darajadagi dasturlash tilida) juda uzoq va tushunish qiyin.

Software crisis va tarkibiy dasturlash tillari

60-yillarning o'rtalaridan o'rtalarigacha bo'lgan davr bugungi kunda "dasturiy ta'minot inqirozi" deb nomlanadi.

Dasturiy ta'minot atamasi 1958 yilda paydo bo'lgan va 10 yildan so'ng ular dasturiy ta'minot inqirozi haqida gapira boshladilar — bu ibora birinchi marta 1968 yilda NATO ilmiy qo'mitasi konferentsiyasida O'qilgan Piter Naur va Brayan Rendellning "dasturiy ta'minot muhandisligi" ma'ruzasida yangradi.

O'nlab yillar davomida 1.0 dasturiy inqiroziga bag'ishlangan bir qator nashrlar paydo bo'ldi. Per Flatten va uning hamkasblarining 1989 yilgi ma'ruzasida keltirilgan baholariga ko'ra, dasturiy ta'minotni ishlab chiqish loyihasini amalga oshirish uchun o'rtacha 18 oy kerak bo'lgan. 1988 yilda Business Week maqolasi mualliflari bu ko'rsatkichni uch yilga teng deb atashganini hisobga olsak, bu



konservativ baho, 1982 yilda tahlilchilar dasturiy loyihalarga besh yil vaqt ketishini ta'kidlashdi.

1994 yilda IBM mutaxassislari tomonidan o'tkazilgan tadqiqot hisoboti tasdiqlandi. barcha dasturiy ta'minot loyihalarining 68 foizi jadvaldan orqada qolmoqda. Shuningdek, ishlab chiqish loyihalari byudjetlarining ortiqcha miqdori 65% ga etishi haqida ma'lumot berilgan. Chiqarilgan, ammo foydalanilmagan dasturiy ta'minot tizimlarini belgilash uchun ular hatto shelfware atamasini — "javondagi dasturiy ta'minot"ni ham o'ylab topishdi.

Standish Group mutaxassislari 2011 yilda chiqarilgan CHAOS Manifesto hisobotida muvaffaqiyatsiz loyihalarining yuqori ulushidan shikoyat qilmoqdalar, ammo ular qo'llagan tahlil metodologiyasi va xulosalari shubha ostiga qo'yildi. Dasturiy ta'minot inqirozi 1.0 allaqachon o'tib ketganga o'xshaydi va dasturiy ta'minotni ishlab chiqish jarayonining ko'plab qo'shimcha yaxshilanishlari tufayli vaziyat hali ham yaxshi tomonga o'zgardi. Amaliy o'zgarishlar oxir-oqibat bugungi kunda dasturiy ta'minot odatda byudjet doirasida ishlab chiqilishiga va texnik talablarga javob berishiga olib keldi. Afsuski, endi 2.0 inqirozi yaqinlashmoqda va rasmda ko'rsatilgandek, uning asosiy sababi so'nggi 50 yil ichida paydo bo'lgan juda katta miqdordagi ma'lumotlardan samarali foydalanadigan, shuningdek qurilmalarning texnik xususiyatlari va foydalanuvchilarining talablarini inobatga olgan holda dasturiy ta'minotni yaratmaslikdir.

Ob'ektga yo'naltirilgan dasturlash

OOP kodni loyihalash, qo'llab-quvvatlash va qayta ishlatishni osonlashtirish uchun ishlab chiqilgan. OOPning asosiy tushunchalari katta va murakkab dasturlarni ishlab chiqishda yordam berish uchun ishlatiladigan inkapsulyatsiya va abstraktsiyadir. OOP-bu dasturlarni tushunish, boshqarish va qo'llab-quvvatlash uchun millionlab kod satrlari dasturlarini yaratish vositasi.

OOP-dan to'g'ri foydalanish kodni tushunish va keyinchalik o'zgartirish uchun osonroq, xavfsizroq va osonroq qilish imkonini beradi. OOP shuningdek, dasturning boshqa qismida yoki hatto boshqa dasturda funksiyalarni jiddiy modifikatsiyasiz qayta ishlatishga yordam beradi. Bu bir xil kodni qayta yozishni va umuman dasturchining ishlashini kamaytirishga yordam beradi.

Strukturaviy va ob'ektga yo'naltirilgan dasturlash

Qaysi dasturlash "yaxshiroq" ekanligi haqidagi munozaralar mantiqiy emas. Dasturlash, u yoki bu shaklda, muammoni hal qilishdan iborat, har qanday dasturlash paradigmasi yordamida har qanday muammoni hal qilishingiz mumkin, ayniqsa ko'plab dasturlash tillari bir nechta paradigmalarda kod yozishga imkon



beradi. Biroq, barcha paradigmlar ma'lum muammolarni teng darajada samarali hal qilishga imkon bermaydi.

Xulosa

Shunday qilib, dasturlash paradigmasi haqidagi munozaralar siz hal qilmoqchi bo'lgan vazifa aniqlanmaguncha mantiqiy emas. Muammoni va uning parametrlarini aniqlaganingizdan so'ng, uni hal qilish uchun qaysi paradigma yaxshiroq ekanligini tushunishingiz mumkin.

Список литературы:

1. Normurodov Ch.B. Mengliyev Sh.A. PHP7 dasturlash tili – O'quv qo'llanma – Termiz: —Xamidi xususiy firmasi, 2020, 218-222 betlar.
2. Musayeva, S. (2022). DESCRIPTION OF MODERN MARKETING RESEARCH METHODS IN THE MARKET ECONOMY. Science and innovation, 1(A5), 33-38 betlar.
3. Vasilev A. N. Python na primerax. Prakticheskiy kurs po programmirovaniyu. — SPb. Nauka i Texnika, 2016. 430-432 st.
4. Yuldasheva, D. (2022). METHODOLOGY FOR DEVELOPING STUDENTS'COMMUNICATIVE COMPETENCE IN ENGLISH (ON THE EXAMPLE OF TOURISM DIRECTIONS). Science and innovation, 1(B7), 1498-1501.
5. Yuldasheva, D. (2021). PHILOSOPHY OF LIFE-AS A DRIVING FORCE. Theoretical & Applied Science, (6), 633-635.
6. Yuldasheva, D. A. (2021). Some essential trends in teaching second language vocabulary. Academic research in educational sciences, 2(6), 782-786.
7. Yuldasheva, D. (2022). The Intensification Of Learning Uzbek Language Using Moodle Technology. Berlin Studies Transnational Journal of Science and Humanities, 2(1.6 Philological sciences).
8. Yuldasheva, D. B. (2021). Approach is the main strategic direction which defines the components of teaching the Uzbek language. Science and World. International scientific journal, (2), 90.
9. Dilnoza, Y. (2019). About Conducting Research on the Subject of Ornithological Terms in Uzbek Language. ANGLISTICUM. Journal of the Association-Institute for English Language and American Studies, 7(7), 53-61.
10. Yuldasheva, D. (2022). THE EMERGENCE OF NEW ECONOMIC WORDS AND PHRASES IN THE LEXICAL SYSTEM OF THE MODERN UZBEK LANGUAGE. 湖南大学学报 (自然科学版), 49(10).