O’ZBEKISTON RESPUBLIKASI AXBOROT TEXNOLOGIYALARI VA KOMMUNIKATSIYALARINI RIVOJLANTIRISH VAZIRLIGI

MUHAMMAD AL-XORAZMIY NOMIDAGI TOSHKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI

NUKUS FILIALI

Dasturiy injiniring kafedrasi
Dasturiy injiniring yo’nalishi

Himoyaga ruhsat etildi
Kafedra mudiri
Utiliev.N
_____
2019 y. «___»_____

Interaktiv matematik funksiyalar grafigini chizish dasturiy majmuasini Java tilida yaratish

BITIRUV MALAKAVIY ISHI

Bitiruvchi : Allaberganov M.

Rahbar: _____

Ilmiy maslahachi: Bobojanov E.

Nukus 2019

O‘ZBEKISTON RESPUBLIKASI AXBOROT TEXNOLOGIYALARI VA KOMMUNIKATSIYALARINI RIVOJLANTIRISH VAZIRLIGI

MUHAMMAD AL-XORAZMIY NOMIDAGI TOSHKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI

NUKUS FILIALI

TELEKOMUNIKATSIYA TEXNOLOGIYALARI VA KASB TA`LIMI FAKULTETI

“DASTURIY INJINIRING” KAFEDRASI

TASDIQLAYMAN

Kafedra mudiri_____

«____» _____ 2019 y.

Allaberganov Maxmudning

Interaktiv matematik funksiyalar grafigini chizish dasturiy majmuasini Java tilida yaratish mavzusidagi bitiruv malakaviy ishiga

TOPSHIRIQ

Bitiruv malakaviy ishining mavzusi

TATU NF-ning «____» _____ 2018 y. buyrig’i bilan tasdiqlandi.

BMI ni topshirish muddati: «____» _____ 2019-yil.

BMIni bajarishga tegishli ma’lumotlar: oquv materiyallari, maruza matn materiallari, ilmiy adabiyotlar va Internet resurs materiyallari.

Bitiruv malakaviy ishining tarkibi:

Kirish

1.Matematik funksiyalar grafigini chizish dasturiy vositalar tahlillari

2.Tekislikda funksiya grafigini chizish dasturiy vositalari.

3.Diagramma va Gistogramma chizish asoslari

4.Java dasturlash tilida interaktiv matematik funksiyalar grafigini chizish dasturiy majmuasini ichlab chiqish.

Xulosa

Foydalanilgan adabiyotlar ro’yxati

Topshiriq berilgan sa’na «____» _____ 2019-yil.

Ilmiy rahbar _____

Topshiriqni oldı _____

Bitiruv malakaviy ishining paragraflari bo'yincha maslahatchilar.

Paragraflar	Ilmiy maslahatchi	Imzo, sana	
		Topshiriqni berdi	Topshiriqni oldi
1	E.Bobojanov		
2	E.Bobojanov		
3	E.Bobojanov		
4	E.Bobojanov		

BMI ishini bajarish rejasi:

№	Bob (paragraf) nomi	Bajarish muddati	Ilmiy rahbar (maslahatchi imzosi)
1.	Matematik funksiyalar grafigini chizish dasturiy vositalar tahlillari	9.02.2019	
2.	Tekislikda funksiya grafigini chizish dasturiy vositalari.	10.04.2019	
3.	Diagramma va Gistogramma chizish asoslari	15.05.2019	
4.	Java dasturlash tilida interaktiv matematik funksiyalar grafigini chizish dasturiy majmuasini ichlab chiqish.	2.06.2019	

Bitiruvchi M.Allaberganov «____»_____2019 -yil.

Ilmiy maslahatchi E.Bobojanov «____»_____2019 -yil.

Mundarija:

KIRISH

Hozirgi vaqtda axborotni qayta ishlash vositalari sifatida kompyuter texnologiyalarining keng tarqalishi jamiyatni axborotlashtirishga axborot texnologiyalari deb ataladigan sohaning rivojlanishiga olib keldi.

Yangi texnologiyalar rivojlangani sari jamiyatdagilar uchun foydali ammallar oshib boradi,chunki odatda hozir har bir sohada axborot texnologiyalariga bo'lgan talab oshib bormoqda ayniqsa meditsina va ta'lim sohalari axborot texnologiyari bilan chambarchas bo'g'liq bo'lib kelmoqda.

Meditsinada axborot texnologiyalarini qo'llash orqali insonning salomatligi yaxshilash foizi ortib kelmoqda, Ta`lim sohasida esa o'quvchilar bilim salohiyatini oshirishda ancha qulaylik yaratadi.

Mavzuning dolzarbligi. Axborot texnologiyalarini ta`lim sohalarida qo'llash bugungi kunda juda muhim bo'lib sanaladi.Axborot texnologiyalarini ta`lim sohalarida qo'llash orqali murakkab masalarni oddiy va o'quvchilar tushunadigan tarzda yechish imkoni osonlashadi.O'quvchilar odatda matematikani o'rganishda biroz qiynalishadi shuning uchun matematikani o'rgatishda vizualizatsiya jarayonlarini qo'lagan holda o'rgatish o'quvchi tushunish uchun ancha oson va qulay bo'ladi.Bunda asosan biror bir matematik funksiyani yechishda uni grafigini tasvirlab berish orqali o'quvchi bu masalaning to'liq moxiyatini anglab yetishi mumkin bo'ladi.

Bitiruv malakaviy ishning maqsadi va vazifalari. Ushbu bitiruv malakaviy ishining maqsadi ta'lim va boshqa murakkab xisob kitoblar amalga oshiriladigan sohalarda axborot texnologiyalaridan foydalangan holda halqilish, Malakaviy ishning vazifasi shundan iboratki, istalgan turdagi matematik funksiyalar grafigini chizib berish va u orqali murakkab masalarga yechim topish. Qo'yilgan maqsadga erishish uchun quyidagi vazifalar belgilab olindi:

- Matematik funksiyalar grafigini chizishni tahlil qilib chiqish;
- Matematik funksiyalar chizadigan dasturiy muhitlar bilan tanishib chiqish;
- Tekislikda grafik chizish usullarini o’rganib chiqish;
- Diagramma va Gistagrammalarni chizishni o’rganib chiqish;
- Java dasturlash tili bilan tanishib chiqish;
- Java dasturlash tiling grafik imkoniyataridan foydalanib dastur tuzish;
- Interaktiv funksiyalar grafigini chizishni Java tilida ishlab chiqish.

Ishning tuzilishi va tarkibi. Bitiruv malakaviy ishi kirish, 2 ta bob va xulosadan iborat. Kirish qismida mavzuning dolzarbligi va maqsadi ochib berildi.

Birinchi bobda matematik funksiyalar grafigini chizish dasturiy vositalar tahlil qilib chiqiladi.

Ikkinchi bobda Java dasturlash tilida interaktiv matematik funksiyalar grafigini chizish va dasturiy majmuasini ishlab chiqiladi.

Xulosa qismida olingan natijalar bajarilgan ishlar tahlil qilinadi.

I-BOB. MATEMATIK FUNKSIYALAR GRAFIGINI CHIZISH DASTURIY VOSITALAR TAHLILLARI

1.1. Matematik funksiyalar grafikini chizish asoslari

Matematikada funktsiya aslida o'zgaruvchan miqdordan boshqa miqdorga bog'liqligini idealizatsiyalash edi. Misol uchun, sayyoralarning pozitsiyasi vaqtning funktsiyasidir. Tarixiy nuqtai nazardan, 17-asrning oxirida tushunchasi infinitesimal hisob-kitob bilan ishlab chiqilgan va 19-asrga qadar ko'rib chiqilgan funktsiyalar farqlanadigan (ya'ni, ular muntazamlik darajasiga ega bo'lgan). Funktsiyaning kontseptsiyasi 19-asrning oxirida to'siq nazariyasi bo'yicha rasmiylashtirildi va bu kontseptsiyani qo'llash sohalarini ancha kengaytirdi.

Matematik nuqtai nazardan qaraladigan bo’lsa funksiya bu biror bir yoki bir nechta o’zqaruvchilar qiymatlarini boshqa bir miqdorga bog’liq tarzda o’zgarishir. Odatda bir nechta qiymatlar orasida bir xil amal bajarishga muhtojlik bo’lsa funksiya yartiladi va shu funksiya yordamida masalani qisqa va oddiy tarzda yechish imkoni mavjud bo’ladi.

Funksiyalar odatda “F(x)” ko’rinishida belgilanadi. Funksiyalar bajaradigan vazifaziga ko’ra bir necha turlarga bo’linadi. Masalan chiziqli funksiya, kvadratik funksiya, integrallashgan funksiyalar va hakazo. Shu funksiyalarni taxlil qilishning ham bir nechta usullari bor bo’lib ularni analitik usul,jadval usul va grafik usullarda tahlil qilish mumkin. Masalani qo’yilgan sharti va yechish oson va qulay bo’ladigan shartni qanoatlantiradigan usul tanlanib funksiyani tahlil qilish mumkin.

Funksiya umumiy holda analitik, jadval, grafik va so‘z usullari bilan berilishi mumkin.

Analitik usul. Ko‘pincha x va y o'zgaruvchilar orasidagi bog'lanish formulalar yordamida ifodalanadi. Bunda argument x ning har bir qiymatiga mos

keladigan funksiyaning y qiymati x ustida analitik amallar — qo‘shish, ayirish, ko‘paytirish, bo‘lish, darajaga ko'tarish, ildizdan chiqarish, logarifmlash va h.k. amallami bajarish natijasida topiladi. Odatda, bunday usul funksiyaning analitik usulda berilishi deyiladi.

Funksiya analitik usulda quyidagi ko‘rinishlarda berilishi mumkin.

1) v=g(x) yoki x=g(y) ko‘rinishdagi formulalar bilan berilgan funksiyalar oshkor ko‘rinishda berilgan funksiyalar deyiladi. Masalan, y=6x-2, $y=x^2+\ln x$ funksiyalar oshkor ko‘rinishda berilgan. Analitik usulda berilgan funksiya bir nechta formulalar vositasida yozilishi ham mumkin, masalan:

$$f(x)=\begin{cases}\cos x, & -\pi\le x\le 0 \quad \text{bo‘lganda},\\ 1, & 0<x<1 \quad \text{bo‘lganda},\\ \dfrac{1}{x}, & 1\le x\le 2 \quad \text{bo‘lganda}.\end{cases}$$

Bu funksiyaning aniqlanish sohasi [-π; 2] bo‘lib, u uchta formula yordamida berilgan.

2) Agar x va y o‘zgaruvchilar qandaydir F(x,y)=0 tenglama bilan bog‘langan, ya‘ni tenglama y ga nisbatan yechilmagan bo'lsa, u holda funksiya oshkormas ko ‘rinishda berilgan deyiladi. Masalan, $x^2 + y^2 – R^2 = 0$ tenglama oshkormas shaklda berilgan funksiyani ifodalaydi, uni y ga nisbatan yechish natijasida ikkita funksiyani hosil qilamiz:

$$y=\pm\sqrt{R^2-x^2}.$$

Balzi bir oshkormas ko‘rinishdagi funksiyalarni y=f(x) (oshkor) ko‘rinishda ifodalash ham mumkin. Har qanday oshkor ko‘rinishdagi y = f(x) funksiyani oshkormas ko‘rinishda yozish ham mumkin: y —f (x)=0.

3) parametrik ko’rinishda, ya’ni:

$$\begin{cases} x = \varphi(t) \\ y = \psi(t) \end{cases} \alpha \le t \le \beta$$

Shu shaklda berilishi. y = f (x) funksiyada x ning y ga mos qo'yilishi parametr deb ataladigan uchunchi bir t o‘zgaruvchining yordamida ifodala- nishi mumkin:

$$\begin{cases} x = \varphi(t) \\ y = \psi(t) \end{cases} \alpha \le t \le \beta,$$

Funksiyalar berilishining eng ko‘p uchraydigan usuli analitik usuldir. Bu usul matematik analizda juda ko‘p ishlatiladi.

Jadval usuli. Ba‘zi hollarda x € X va y € Y o‘zgaruvchilar orasidagi bog’lanish formulalar yordamida berilmasdan, balki jadval orqali berilgan bo‘lishi ham mumkin. Masalan, t — yanvar oyining birinchi dekadasi (10 kunligi) kunlari nomeri bo'lsa, T — shu nomerli kuni soat 16^{00} da Urganch shahrida kuzatilgan havo haroratini bildirsin, natijada quyidagi jadvalga kelamiz:

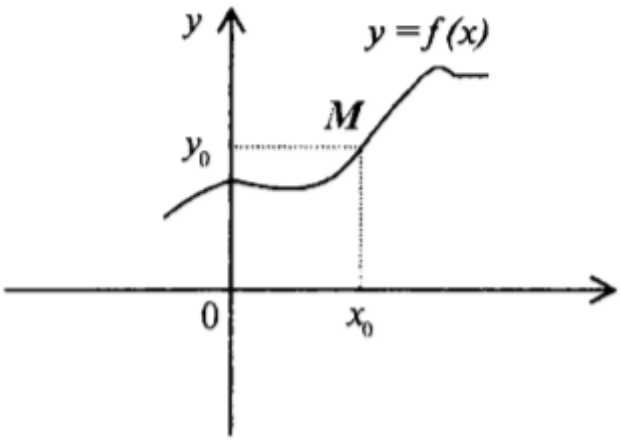
t	1	2	3	4	5	6	7	8	9	10
T	−3°	−5°	+2°	+5°	+1°	0°	−2°	−5°	−3°	−1°

bunda t — argument, T — funksiya bo‘ladi. Bog’lanishning bunday berilishi jadval usulda berilishi deb ataladi. Bu usuldan ko‘pincha miqdorlar orasida tajribalar o‘tkazish jarayonida foydalaniladi.

Jadval usulining qulayligi shundan iboratki, argumentning u yoki bu aniq qiymatlarida, funksiyani hisoblamasdan, uning qiymatlarini aniqlash mumkin. Jadval usulining qulay bo‘lmagan tomoni shundan iboratki, argumentning o‘zgarishi bilan funksiyaning o‘zgarish xarakterini to‘liq aniqlab bo‘lmaydi.

Grafik usuli. xOy koordinata tekisligida x ning X to‘plam (X=D(f)) dan olingan har bir qiymati uchun M(x,y) nuqta yasaladi, bunda nuqtaning abssissasi x, ordinatasi y esa funksiyaning x ga mos kelgan qiymatiga teng. Yasalgan

nuqtalami tutashtirsak, natijada biror chiziq hosil boiadi, hosil bo‘lgan bu chiziqni berilgan funksiyaning grafigi deb qaraladi (1.1- chizma).



1. 1.1-chizma

Tekislikning (x, f(x)) kabi aniqlangan nuqtalaridan iborat ushbu

{(x, /(x))}={(x/(x)):x € X, y=f(x) € Y}

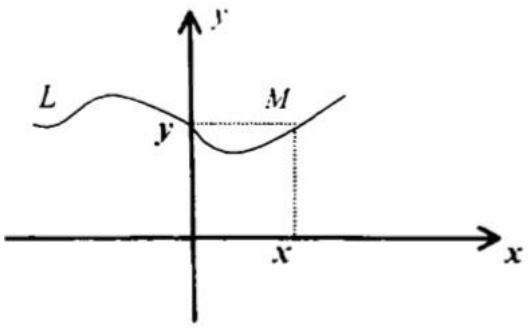
to‘plam, funksiyaning grafigi deb ataladi.

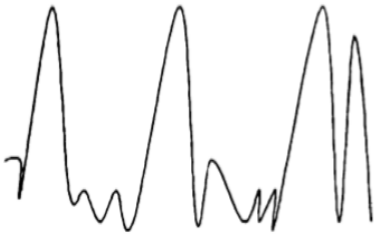
xOy tekisligida shunday L chiziq berilgan bo‘lsin, Ox o‘qda joylashgan nuqtalardan shu o‘qqa o‘tkazilgan perpendikular L chiziqni faqat bitta nuqtada kesib o‘tsin. Ox o'qdagi bunday nuqtalardan iborat to‘plamni X orqali belgilaymiz. X to‘plamdan ixtiyoriy x ni olib, bu nuqtadan Ox o‘qiga perpendikular o‘tkazamiz. Bu perpendikularning L chiziq bilan kesishgan nuqtasini y bilan belgilaymiz. Natijada X to‘plamdan olingan har bir x ga yuqorida ko'rsatilgan qoidaga ko‘ra bitta y mos qo‘yilib, funksiya hosil bo’ladi. Bunda x va y o‘zgaruvchilar orasidagi bog'lanish L cliiziq yordamida berilgan bo‘ladi (1.2-chizma). Odatda funksiyaning bunday berilishi uning grafik usulda berilishi deb ataladi.

Funksiyaning grafik usulda berilishi ilmiy tadqiqotlarda va hozirgi zamon ishlab chiqarishi jarayonlarida keng qo'llaniladi. Masalan, tibbiyotda uchraydigan elektrokardiogramma grafigi—yurak muskullaridagi tok impulslarining vaqt

bo‘yicha o'zgarishini ko‘rsatadi. Bu grafik analitik tarzda yozilishi shart bo‘lmagan biror y =f(x) funksiyaning grafigidir, bu funksiyaning formulasi shifokor uchun unchalik qiziqarli emas (1.1.3- chizma).

Funksiyaning grafik usulda berilishining kamchiligi shundan iboratki, argumentning sonli qiymatida berilgan funksiyaning aniq ko'rinishini har doim topib bo'lavermaydi, lekin bu usulning boshqa usullardan afzalligi uning ta‘siri yaqqol ko‘zga ko‘rinib turishidadir.





1.1.2-chizma **1.1.3-chizma**

So‘zlar orqali ifodalanadigan usul. Bu usulda (x€X, va y€Y) o‘zgaruvchilar orasidagi funksional bog‘lanish faqat so‘zlar orqali ifodalanadi.

1- misol. Har bir ratsional songa 1 ni, har bir irratsional songa 0 ni mos qo‘yish natijasida ham funksiya hosil bo‘ladi. Bu funksiya, odatda, Dirixle funksiyasi deyiladi va D(x) kabi belgilanadi:

$$D(x) = \begin{cases} 1, \text{agar } x \text{ ratsional son bo‘lsa,} \\ 0, \text{agar } x \text{ irratsional son bo‘lsa.} \end{cases}$$

2- misol. f — har bir x haqiqiy songa uning butun qismi [x] ni mos qo‘yuvchi qoida bo‘lsin. Demak, f : x —»[x] yoki y= [x] funksiyaga ega bo‘lamiz.

3- misol. f — har bir haqiqiy x songa uning kasr qismi {x} ni mos qo'yadigan qoida bo‘lsin, ya‘ni f:x —» {x}. Bu holda biz y={x} funksiyaga ega bo‘lamiz.

Argumentning funksiya ma'nosini yo‘qotmaydigan (ya‘ni cheksiz yoki mavhumlikka aylantirmaydigan) hamma qiymatlari to‘plami shu funksiyaning aniqlanish sohasi deyiladi. Agar funksiya jadval shaklida berilsa, uning aniqlanish sohasi x ning jadvalda ko‘rsatilgan qiymatlaridan iborat bo‘ladi. Agar funksiya grafik shaklda berilsa, uning aniqlanish sohasi grafikdan ko'rinib turadi. Funksiya analitik shaklda berilganda esa x ning funksiyani aniqlaydigan formula ma'noga ega bo‘ladigan qiymatlari to‘plami shu funksiyaning aniqlanish sohasi bo'ladi. Funksiyaning aniqlanish sohasini topish vaqtida formulani boshqa ko'rinishga keltirish tavsiya etilmaydi.

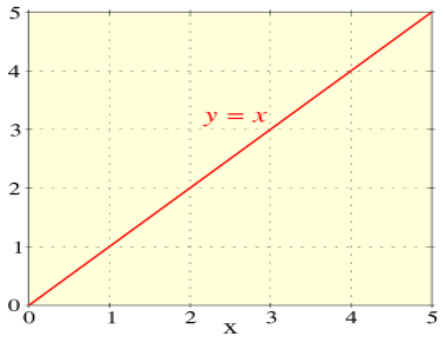
Funktsiya - bu funktsiyaning ta'sir doirasi X ning har bir elementini x ga qo'shadigan jarayon yoki munosabat, ya'ni boshqa funktsiyaning Y (ehtimol bir xil to'siq) ning bitta elementiga, funktsiyaning kodomainasiga bog'liqdir. Funktsiya f deb ataladigan bo'lsa, bu munosabat y = f (x) (x) o'qiladi, element x funksiyaning argumenti yoki kiritilishi va y funksiyaning qiymati, chiqishi yoki f tomonidan f (x) ko'rinishi kirishni ifodalash uchun ishlatiladigan belgilar funktsiyaning o'zgaruvchisidir (ko'pincha f ning o'zgaruvchining x funktsiyasidir).

Funktsiya barcha juftliklar majmuasi bo'lgan (x, f (x)) grafigi bilan ifodalanadi. Domen va kodomain raqamlar majmuasi bo'lganda, har bir juftlik tekislikdagi nuqta karteziya koordinatalari deb hisoblanishi mumkin. Umuman olganda, ushbu nuqtalar funktsiyaning grafigi deb ataladigan bir egri hosil qiladi. Bu hamma joyda keng tarqalgan funktsiyaning foydali tavsifi. Misol uchun, narx ko'rsatkichlari va birja indekslarining evolyutsiyasini ifodalash uchun, gazetalarda ko'pincha funktsiyalarning grafikalari ishlatiladi.

Matematikada f funktsiyasining grafikasi rasmiy ravishda barcha buyruqlar juftlarining (x, f (x)) majmui bo'lib, bu x funktsiya funktsiyasi doirasiga kiradi.

Ikkita o'zgaruvchining funktsiyasi bo'lib, domen ikki juftdan tashkil topgan (x, y) funktsiyalar bo'lsa, grafik barcha tartiblangan uchburchaklarning to'plamiga ((x, y, f (x, y)) aniqlanishi mumkin. Ikkita real o'zgaruvchining doimo aniq qiymatli funksiyasi, grafika bir sirtdir.

Funktsiyaning grafigi tushunchasi munosabatlar grafigiga umumlashtiriladi. Bir munosabatning grafigi birinchi o'zgaruvchining x funktsiyasini anglatishini tekshirish uchun, vertikal chiziq testidan foydalaniladi. Grafika ikkinchi o'zgarmaydigan y funktsiyasini anglatadimi yoki yo'qligini tekshirish uchun gorizontal chiziq testidan foydalaniladi. Funksiya teskari bo'lsa, teskari grafika asl funktsiyaning grafigini y = x yo'nalishida aks ettirish orqali topiladi.



1.1-rasm.y = x grafigi

1.2. Tekislikda funksiya grafigini chizish dasturiy vositalari

Matematikani o’rganish matematik moslamalarni va protseduralarni ko'rish va namoyish etish bilan chambarchas bog'liq.

Ta'limdagi texnologiyadan foydalanish an'anaviy ta'limdan ko'ra o'rganishni yaxshilaydi.Matematikani o’rganishda qo’llaniladigan bir qancha dasturiy vositalar bordir, ular yordamida matematikani tushunish yanada yaxshi bo’ladi ya`ni bu dasturiy vositalar yordamida o’quvchilar matematik xisob kitoblarni vizual tarzda ko’rib bajarishlari mumkin bo’ladi. Matematik xisob

kitoblarni bajaradigan dasturiy vositalarga misol qilib Microsoft Mathematics, Matlab,Maple va boshqa dasturiy vositalarni olishimiz mumkin. Microsoft Mathematics dasturi Microsoft kompaniyasi tomonidan tayyorlangan bepul dasturiy vosita hisoblanadi, bu dasturda hisoblash tizimi matematik ifodalarga asoslangandir.

Microsoft Mathematics – bu matematik hisoblash dasturlari bo'lib o'quvchilarga muammolarni hal qilishda yordam berish uchun foydalidir. Bu dastur yordamida Lineer algebra, statistika, matematika va trigonometik xisob kitoblarni amalga oshirish mumkin.

Microsoft Mathematics dasturi ta`limida matematikani o'rganish asosiy maqsadlardan biri va barcha talabalarning matematikani tushunishdagi muvaffaqiyatini ta'minlashdir. Matematikani o'rganish eng murakkab masalalardan biri hisoblanadi va ta'limning muammoli jihatlari hisoblanadi. Shu bilan birga,eng muhimi matematikani kundalik hayotda keng qo'llaniladigan bilim sifatida fanga o'qitish hisobga oladigan va boshqa ko'plab ilm-fan sohalarida. Matematika - har bir sohaning kontseptsiyalarini tahlil qilishning asosiy vositasi inson hayotining yo'nalishi. Shuning uchun o'qituvchilarni o'rgatish tizimini rivojlantirishga alohida e'tibor berish kerak.

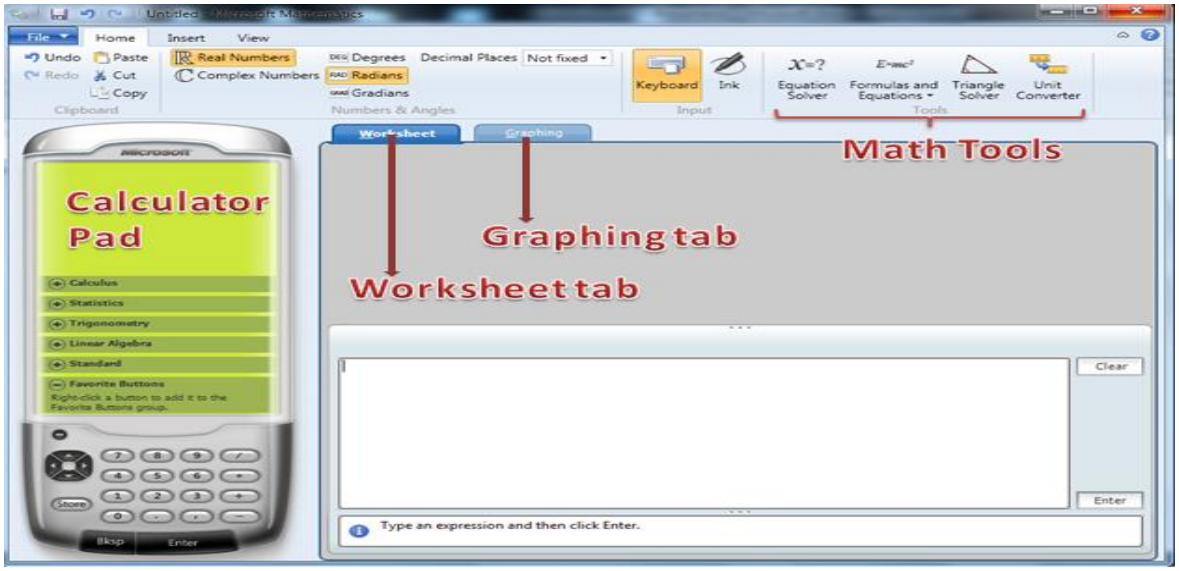
Ko'pgina talabalar matematik tushunchalar bilan shug'ullanish uchun qiyinalishadi. Vizualizatsiya yordamida qo'llab-quvvatlanadigan tadbirlar matematikani o'rganishni yaxshilaydi.

Matematikani o'rganishga yordam berish uchun grafik tasvir yaratish o'quvchilar tushunishni yanda yaxshilaydi. Shuning uchun matematikani o'rganishda Microsoft Mathematics daturidan foydalanish ancha qulay va tushunarli bo'ladi. Bu dasturda matematik xisob kitoblarni yechibginaqolmay balki ularni grafiklarini chizib ham ko'rish imkoniyati mavjudir. Bu esa yuqorida takidlanganidek matematikani o'rganishda grafik tasvirdan foydalanish tushunish uchun qulaydir.

Matematika deyarli hamma o'quv dasturlarida muhim rol o'ynaydi misol uchun muhandislik, fan, biznes, iqtisod, informatika va axborot kabi fanlarni o'z ichiga oladi.

Dasturda matematika tushunchalari sodda, mantiqiy va ierarxik tarzda eng sodda tarzda joylashtirilgan.Boshqacha qilib aytganda, kontseptsiyani tushunish va o'zlashtirish mahorat talab qiladi.

Microsoft Mathematics - Microsoft komponiyasi tomonidan ishlab chiqilgan bepul dasturiy ta'minot. Ushbu dastur matematik masalarni yozish, hisoblash va manipulyatsiya qilish,2D, 3D grafik tasvirlarni hosil qilish va animatsiyani ifodalash va grafik vizualizatsiya qilish oson amalga oshirishni ta'minlaydi.



1.2.1-rasm. Microsoft Mathematics daturining interfeysi

Microsoft Mathematics oynasida to'rtta asosiy qism bor:

1. Kalkulyator paneli(Calculator Pad);
2. Ishchi yorlig'i(Worksheet tap);
3. Tasvirlash yorlig'i(Graphing tap);
4. Matematika vositalari(Math tools);

Kalkulyator paneli(Calculator Pad) – bu qism matematik hisoblash uchun tugma guruhlaridan iborat. O'rta maktab va o'rta maktab matematikasida ishlatiladigan standart tugmalardan tashqari,matematika, statistika, trigonometriya va lineer algebra kabi tugmalar ham mavjud.

Ishchi yorlig'i(Worksheet tap) – bu qismda Microsoft Mathematics dasturini ochishda ko'rsatuv yorliq paydo bo'ladi. U erda ko'pgina kirish va chiqish ko'rsatiladi va pastki qismida matn yozish joyida klaviaturadan yozish imkoni mavjud.

Tasvirlash yorlig'i(Graphing tap) – bu qism grafikalar yaratish uchun ishlatiladi. Bundan tashqari tenglama, funktsiyalar, tengsizliklar, ma'lumotlar to'plamlari va parametrik tenglamalar uchun ishlatiladigan panellalar mavjud.

Matematika vositalari(Math tools) – bu qism quyidagilarni o'z ichiga oladi:

- Tenglama hisoblovchi - tenglamalarni yoki tenglama tizimlarini echish uchun ishlatiladigan vosita;
- Formulalar va tenglamalar – matematika fanida tez-tez ishlatib turadigan formulalarni o'z ichiga oladi;
- Triangle Solver - uchburchakning burchak o'lchovlari va yon uzunligini topadigan vosita;
- Birlik ishlab chiqarish vositasi - bir o'lchov birligini boshqasiga aylantirish uchun vosita;

Microsoft Mathematics dasturida asosiy matematik hisob-kitoblarni qanday bajarish kerakligini ko’rib chiqamiz

Raqamli hisob-kitoblarda ishlatiladigan Microsoft Mathematicsning uchta qismi mavjud:

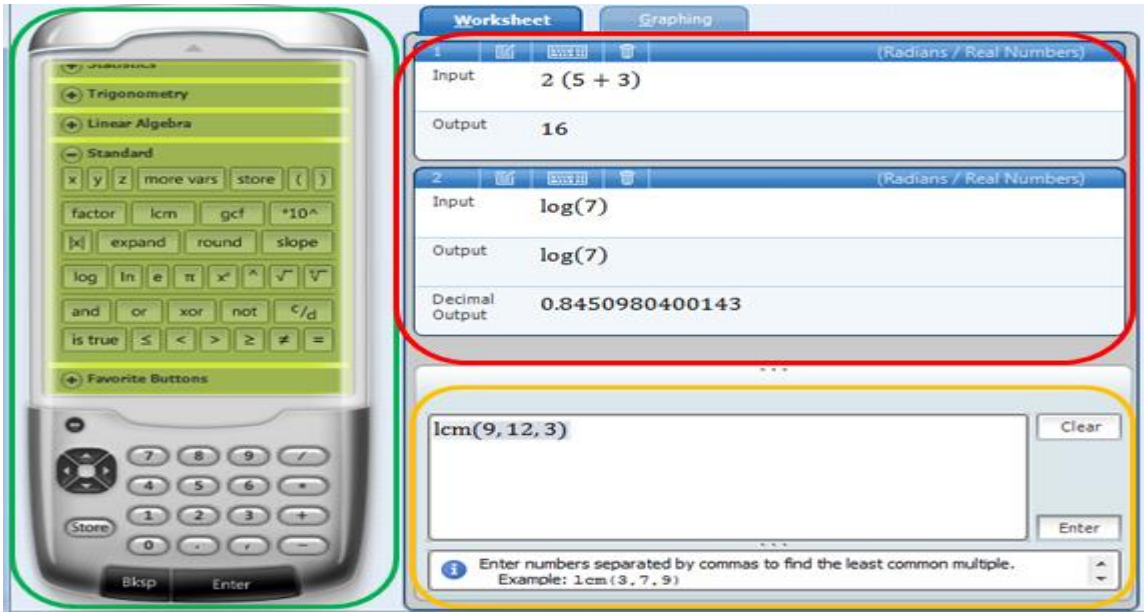
1. Buyruqlar tugmalari joylashgan kalkulyator paneli (yashil quti);
2. Buyruqlar yoziladigan kirish qutisi (sariq quti);

3. Kirish joyi, qadam hisob-kitoblari (mavjud bo'lsa) va hisoblash natijalari ko'rsatiladi. Kiritish matn qutisi va chiqarish qutilari ish varag'i ko'rinishida joylashgan.

Quyidagi hisob-kitoblarni bajarish uchun Microsoft Mathematicsni ochiladi.

1-misol: 2 (5 + 3) ni hisoblash

- ✓ Ishchi yorlig'i(Worksheet tap) sahifasida ishlanilayotganiga ishonch hosil qilinadi;
- ✓ Kursorni hisoblash uchun matn maydoniga kiritiladi (sariq rangli qutiga qarang);
- ✓ Keyin 2 (5 + 3) ni kiritiladi yoki oyna chap pastki qismida joylashgan kalkulyator klaviaturasidan foydalaniladi;
- ✓ Klaviaturadan ENTER tugmasini bosiladi, yoki kalkulyator pastki qismidagi ENTER tugmasini bosiladi;
- ✓ Natija quyidagicha ko’rinadi.

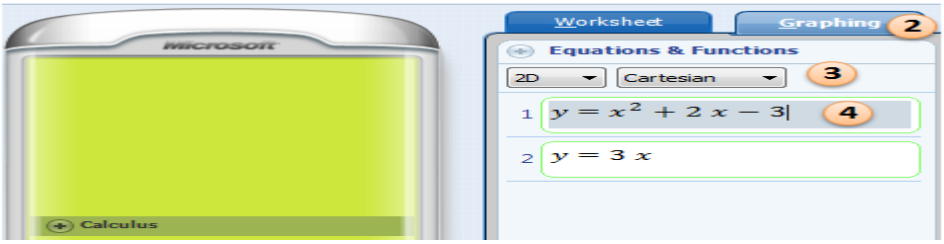


1.2.2-rasm. Microsoft Mathematics daturida asosiy xisob-kitoblar.

Microsoft Mathematics dasturida 2 va 3 o'lchamli kartezyen grafikalar va 2 o'lchovli qutb grafikalarini qanday yaratishni va grafika oynasining parametrlarini qanday qilib o'zgartirishni ko’rib chiqamiz.

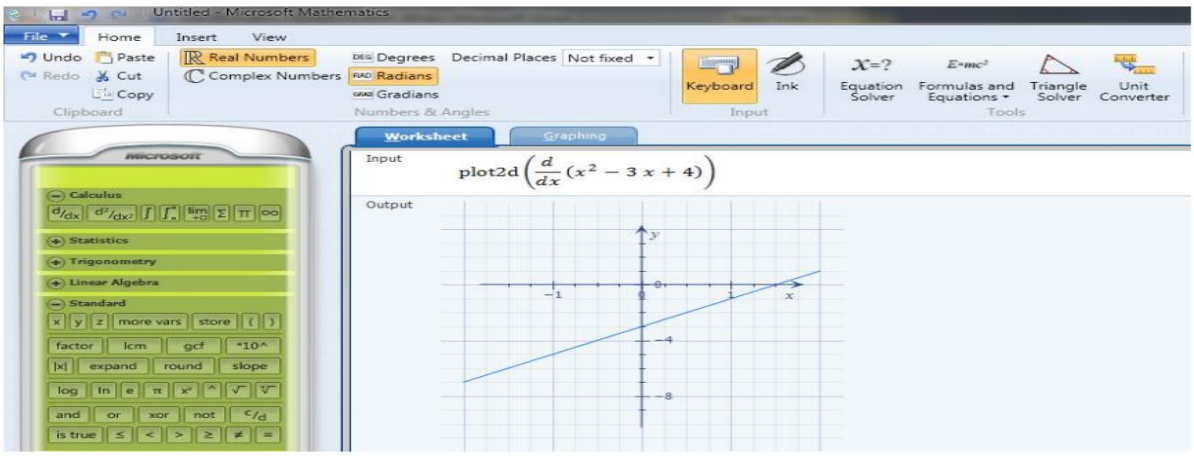
1. Microsoft Mathematics ochiladi.

2. Tasvirlash yorlig'ini tanlanadi.
3. Y = x^2 + 2x - 3 va y = 3x funksiya yoziladi. Ustun uchun ^ belgisini ishlatiladi.
4. Tenglama kiritilgandan so'ng **Graphing** tugmachasi bosiladi.



2.3-rasm. Tenglamani kiritish ketma-ketligi.

Tenglama va funktsiyalar ostida 2D va Cartesianning tanlanganligiga ishonch hosil qilinadi.



1.2.4-rasm. 2D da chizilgan grafik

Grafika ko'rsatilgandan so'ng, quyidagilarni bajarish uchun asboblar panelidagi tugmalardan foydalanishingiz mumkin. Asboblar panelidagi Tasvirni nolga o'rnatish oynasini ochish orqali grfika oynasini tozalash mumkin.

Microsoft Mathematics dasturi yordamida matematik maslarni yechishi oson tarzda amalga oshirsh mumkin va ularning grafiklarini chizib ko’rgan ham matematikani tushunish uchun ancha qulaylik yaratadi.Matematik funksiyalarni yechish va ularning grafikini tasvirovchi boshqa bir qancha dasturiy vositalar ham mavjud bo’lib ularning ichida foydalanish anch oson bolgan dasturiy vosida bu Matlab dasturi hisoblanadi.

Matlabda oddiy hisoblashlarni ko’rib chizamiz

Matlab dasturiy vositasininng tizimi shunday ishlab chiqilganki, hisoblashlarni, foydalanuvchi dasturini tayyorlamasdan to‘g‘ridan-to‘g‘ri bajarish imkoni mavjud. Bunda Matlab super kalkulyator vazifasini bajarib, qatorli komanda rejimida ishlayoladi. Misol uchun: >>2+3, ans=5; >>2*3, ans=6 va xokazo.

Matlab tizimida ishlash muloqotli (dialogli) tavsifga ega bo‘lib, “savol berildi – javob olindi” qoidasi bo‘yicha ishlanadi. ya’ni foydalanuvchi klaviatura yordamida hisoblanishi lozim bo‘lgan ifodani kiritadi, tahrir qiladi (agar lozim bo‘lsa) va kiritishni ENTER klaviaturasini bosib yakunlaydi.

Umuman olganda, ma’lumotlarni kiritish va hisoblashlarni amalga oshirish quyidagicha amalga oshiriladi:

- Boshlang‘ich ma’lumotlarni kiritishni ko‘rsatish uchun “>>” belgisidan foydalaniladi;
- Ma’lumotlar odatda oddiy yozuvli tahrir yordamida kiritiladi;
- Biror bir ifoda hisoblash natijasini blokirovka qilish uchun mazkur ifodadan keyin - ; (nuqta vergul) qo‘yiladi;
- Hisoblashlar natijasini ko‘rsatuvchi o‘zgaruvchi aniqlanmagan bo‘lsa, u holda Matlab tizimi bunday o‘zgaruvchi deb *ans*oladi;
- O‘zlashtirish amali sifatida juda ko‘plab dasturlash tillari kabi : = belgi emas, balki matematikadagi oddiy = ni o‘zi olinadi;
- Sozlangan funksiyalar (masalan, sin) yozma harflar bilan yoziladi hamda ularning argumentlari oddiy qavslar ichida yoziladi;
- Hisoblashlar natijasi yangi qatorda >> belgisiz chiqadi;
- Muloqot “Savol berildi – javob olindi” ko‘rinishida amalga oshadi.

Ma’lum bo’lganidek juda ko‘plab matematik tizimlarda, agar u son bo‘lmasa, u holda sin(v) va exr(v) ifodalarni hisoblab bo‘lmaydi, ya’ni tizim

bunday ifodalarni xato deb beradi. Matlabda esa agar berilgan o‘zgaruvchi vektor bo‘lsa, natija ham mazkur o‘lchamdagi vektor bo‘ladi, agar matritsa bo‘lsa, natija ham matritsa bo‘ladi.

Komandali rejimda asosan bir qatordagi belgilarning maksimal soni – 4096, m – fayllarda esa chegaralanmagan. Barcha matematik tizimlarning markaziy tushunchasi bu matematik ifodalardan tashkil topgan. Ma’lum bo’lishicha, matematik ifodalar ustida amallar bajarilayotganda, asosan ularning sonli qiymatlaridan foydalaniladi (kam holatlarda belgi ko‘rinishlaridan ham foydalaniladi).

Matlabham matematik tizim bo‘lgani uchun bu erda ham asosiy tushuncha matematik ifodalardir. Matlabda matematik ifodalarni ifodalashni qarab chiqaylik. Matlabda ifodalar bir qator ko‘rinishida ifodalanib, sonlarni butun qismlarini ajratish uchun verguldan emas balki nuqtalardan foydalaniladi. Quyida ba’zi bir ifodalarni Matlab va oddiy matematikadagi ifodalanishini ko‘rib chiqamiz:

Matlabda	Matematikada
2+3	2+3
2^3*sqrt(y)/2;	23√y/2
2.301*sin(x);	2,301sin(x)
4+exp(3)/5;	4+e3/5

Matematik ifodalar asosan sonlar, konstantalar, o‘zgaruvchilar, operatorlar, funksiyalar va turli xil maxsus belgilar ustiga quriladi. Ilgari aytib o‘tganimizdek, nuqta vergul, ya’ni ; belgi natijani chiqishini blokirovka qiladi, ammo *ans* maxsus o‘zgaruvchi yordamida natijani olishimiz mumkin.

Son – Matlab tilining eng oddiy ob’ektlaridan biri bo‘lib, u miqdoriy ma’lumotlarni ifodalab beradi. Sonlarni konstanta deb hisoblash mumkin.Sonlar

butun, kasr, fiksirlangan va suzuvchi nuqtali bo‘lishi mumkin. Ularni yaxshi ma’lum bo‘lgan ilmiy shaklda, ya’ni mantissa va son tartibini ko‘rsatgan holda ifodalash mumkin.

```
0
-3
2.301
123.456e-24
-234.456e10
```

yuqoridan ko‘rinib turibdiki, mantissadan sonning butun qismi kasr qismidan, juda ko‘plab dasturlash tillarida qabul qilinganidek, vergul orqali emas, balki nuqta orqali ajratiladi. Son tartibini mantissadan ajratish uchun ular orasiga e belgisi qo‘yiladi. “+” ishora sonlar oldiga qo‘yilmaydi, “-” ishora esa qo‘yiladi va uni unar minus deb nomlanadi. Sonlarda belgilar orasiga probel (bo‘sh joy) qo‘yish ruxsat etilmaydi. Bundan tashqari sonlar kompleks bo‘lishi mumkin: z=Re(z) + Im(z)*i. Bunday sonlar Re(z) haqiqiy va Im(z) mavhum qismga ega bo‘linadilar. mavhum qism kvadrat darajasi -1 ga teng bo‘lgan, *i* va *j* ko‘paytuvchilarga ega bo‘ladi:

```
3i
2j
2+3i
-3.141i
-123.456+2.7e-3i
```

real (z) funksiya kompleks sonning butun qismini, image(z) – esa mavxum qismini ajratib beradi. Kompleks sonning modulini (kattaligini) abs(z) funksiya, fazasini angle(z) funksiya hisoblab beradi.

Misol uchun:

```
>> i
Ans=0+1.000i
>>z=2+3i
Z=2.000+3.000i
>>abs(z)
Ans=3.6056
>>real(z)
```

```
Ans=2
>>Imag(z)
Ans=3
>>angle(z)
Ans=0.9828
```

Matlabda grafik chizishning imkoniyatlarini ko’rib chiqamiz.

Matlab bu asosan matematik xisob kitoblarini amalga oshiruvchi dasturiy vosita bo’lib unda matematik funksiyalar grafiginiham chizish imkoni bor. Chunki Matlab ikki o‘lchovli grafika asosida ishlaydi. Matlab tizimining eng katta xususiyatlaridan biri, unda grafik chizish imkoniyatini mavjudligidir.

Matlabda grafiklarni har xil koordinata sistemalarida qurish mumkin. Bulardan to‘g‘ri burchakli dekart koordinatalari sistemasi, polyar koordinatalari, sferik vassilindrik sistemalarni keltirish mukin. bundan tashqari koordinatalarni bir sistemadagi ko‘rinishidan boshqa ko‘rinishga o‘tkazish mumkin.

Matlab yordamida ikki vektor grafigini chizishning eng sodda va umumiy komandalari bor bo’lib ular quyidagicha yoziladi.

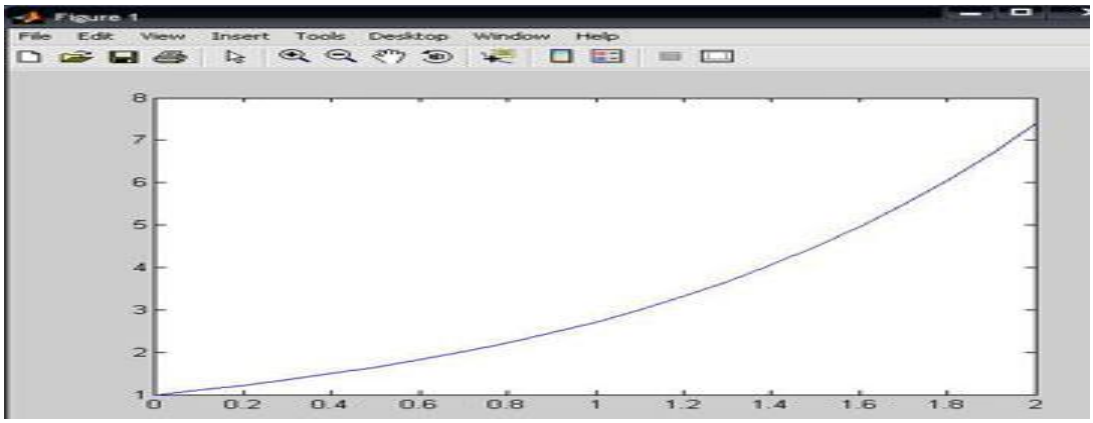
- plot(x,y) x va y vektorlarning dekart tekisligidagi grafigini chizishda ishlatiladi;
- plot(y) y ning y - vektor elementlari nomerlarga nisbatan grafigini hosil qiladi;
- semilogx(x,y) “x”ni logarifmi grafigini “ y” ga nisbatan chizadi;
- semilogy(x,y) “x”ning grafigini “y” ning logarifmiga nisbatan chizishda ishlatiladi;
- loglog(x,y) “x”ni logarifmini “y” ni logarifmiga nisbatan grafigini chizadi;
- grid -koordinatalar sistemasida to‘rni paydo qiladi;
- title (‘matn’)- grafik tepasiga matn yozishda ishlatiladi;
- xlabel (‘matn’) - “matn”ni “x” o‘qi ostiga yozishda ishlatiladi;

- ylabel (‘matn’)- “matn”ni “ y ” o‘qining chap tomoniga yozishda ishlatiladi;
- text(x,y,’matn’)- “matn”ni (x, y) nuqtaga yozishda ishlatiladi;
- polar(theta, r)- r va theta vektorlarning polyar koordinatalar sictemasida grafigini hosil qiladi (theta radianlarda beriladi);
- bar(x) yoki stairs(x)- “x” vektorning gistogrammasini hosil qiladi;
- bar(x,y) yoki stairs(x,y)-“u” vektor elementlarini gistogrammasini “x” vektorning elementlariga mos to‘plamga joylashtirib yasaydi;

Ma’lum bo’lganidek, dekart koordinatalar sistemasida grafik chizish (x, y) juftligini qiymatlarini aniqlab, hosil bo‘lgan nuqtalarni kesmalar bilan tutashtirish orqali yaratiladi.bundan ko’rinib turibtiki (x, y) juftliklar soni qanchalik ko‘p bo‘lsa grafik ham shunchalik silliq va aniqroq bo‘ladi. Juftliklar avvaldan berilgan bo‘lishi yoki ma’lum funksiyaning argumenti va qiymatlaridan hisoblab hosil qilinishi yoki tajriba o‘tkazish natijasida olingan bo‘lishi mumkin.

Misol uchun $y=e^x$ funksiyaning $x\epsilon[0,2]$ sigmentdagi grafigini chizish kerak bo‘lsa, unda quyidagi matlab komadalari ketma-ketligidan foydalaniladi:

```
>>x=0:.1:2;
>>y=exp(x);
>> plot(x,y)
```



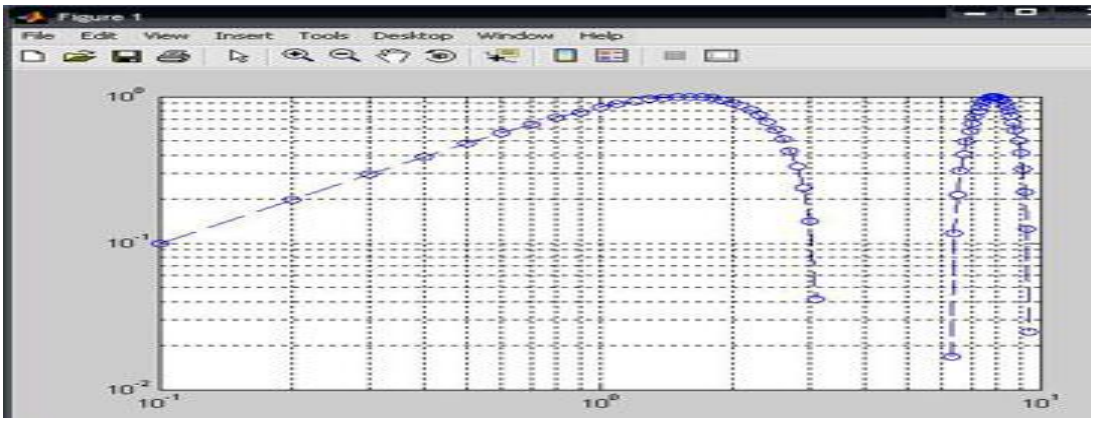
1.2.5-rasm. $y=e^x$ funksiyaning $x\epsilon[0,2]$ sigmentdagi grafigi.

rlot(x,y) – buyrug’i grafik oynasini ochadi va unda kerakli funksiya grafigini hosil qiladi.

Ko‘pgina holatlarda grafik komandalar M-faylga joylashtiriladi (Ishchi fayl yoki fayl funksiyalar). Bu usul xatoliklarni to‘g‘rilash uchun yaxshi imkoniyat beradi.

Quyidagi misollarni ko’rib chiqamiz:

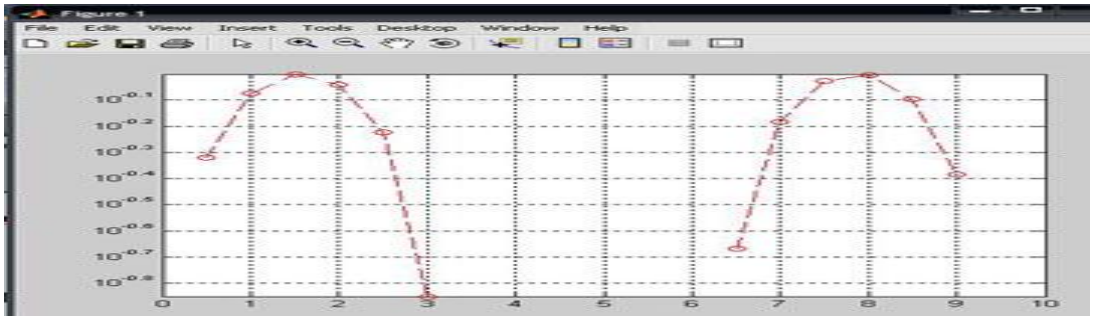
Misol sharti shundan iboratki bunda %x ni logarifmini sin(x) ni logarifmiga nisbatan chizilgan grafigini (x=0:.1:10;loglog(x,sin(x),’--ob’);grid on) chizish.



1.2.6-rasm. %x ni logarifmini sin(x) ni logarifmiga nisbatan chizilgan grafigi

Bu yerda ‘--’ -liniya turi, ‘0’-aylana tugun nuqta turi, ‘b’-havorang liniya rangi.Endi boshqa bir tur grafik funksiyadan foydalanib ko‘ramiz:

```
>> x=0:0.5:10;
>> semilogy(x,sin(x),'--or')
>> grid
```



1.2.7-rasm. grafik chiziqlarini rangini, turini, tugun nuqtalarini ko‘rsatish.

Bu misollardan ko‘rinib turibdiki, matlab tizimida grafik chiziqlarini rangini, turini, tugun nuqtalarini ko‘rsatish va boshqa imkoniyatlar mavjud.

Har bir masalani yechishni bir nechta usuli bo’lgani kabi matematik funksiyalarni chizishda foydalaniladigan dasturiy fositalarning ham turlari bo’r ularda masalani hal qilish bir biridan farq qiladi masalan Maple dasturida matematik funksiyalarni chizish grafik tasvirini chiroyliroq ko’rinishda aks etirb beradi.

Mapleda dasturida funksiya grafigini yasashni ko’rib chiqamiz.

Mapel dasturi ikki o`lchovli grafikaga asoslanib ishlaydi.Biror bir funksiya grafigini chizish uchun **plot** komondasidan foydalaniladi shuning uchun plot komondasi va uning parametrlari ko’rib chiqamiz.

Matematik funksiyaning grafigini (Ox o`qi bo`yicha $a \le x \le b$ intervalda va Oy o`qi bo`yicha $c \le y \le d$ intervalda) hosil qilish uchun **plot** komondasi ishlatiladi. Uning umumiy ko`rinishi quyidagicha bo’lib **plot(f(x), x=a..b, y=c..d, parametrs)**, bu yerda **parametrs** qismi – tasvirni boshqarish imkoniyatlari. Agarda parametrlari ko`rsatilmay qolsa jimlik bo`yicha o`rnatishdan foydalaniladi. Shu bilan bir qatorda tasvirlarga tuzatishlar kiritish vositalar paneli orqali ham amalga oshiriladi. **plot** buyrug`ining asosiy xususiyatlari:

1) **title=”text”** text - bu tasvirning sarlavhasi, lotin harflari bilan probelsiz yoziladi, qo`shtirnoq belgisini yozmasa ham bo`ladi.

2) **coords=qutb** – bu polyar koordinatani o`rnatish parametri.

3) **axes** – koordinata o`qlari turlarini o`rnatish parametri: axes=NORMAL – oddiy o`qlardir; axes=BOXED – ramkada shkalali grafikadir; axes=FRAME – rasmning pastki chap burchagi markazi bo`lgan o`qlardir; axes=NONE – o`qsiz bo’lan parmeter.

4)**scaling** – tasvir masshtabini o`rnatish parametri: scaling=CONSTRAINED – o`qlar bo`yicha bir xil masshtabni o’rnatish; scaling=UNCONSTRAINED – grafik oyna o`lchovi bo`yicha masshtablanadi.

5) **style=LINE(POINT)** – chiziqlar (yoki nuqtalar) bilan chiqarish parametri.

6) **numpoints=n** – grafikaning hisobga olinadigan nuqtalarini (jimlik qoidasi buyicha n=49) o’rnatish.

7) **color** – chiziq rangini o`rnatish parametri: ranglar ingilscha nomlar bilan yoziladi, masalan: yellow – sariq va h.

8) **xtickmarks=nx** va **ytickmarks=ny** – mos ravishda , Ox va Oy o`qlari bo`yicha belgilar sonini aniqlaydi.

9) **thickness=n, n=1,2,3…** - chiziq qalinligini o’rnatish (jimlik buyicha n=1).

10) **linestyle=n** – chiziq turini aniqlash: uzluksiz, punktirli va h. (n=1 – uzluksiz).

11) **symbol=s** – nuqtalar orqali hosil buladigan belgi turini aniqlash: BOX, CROSS, CIRCLE, POINT, DIAMOND.

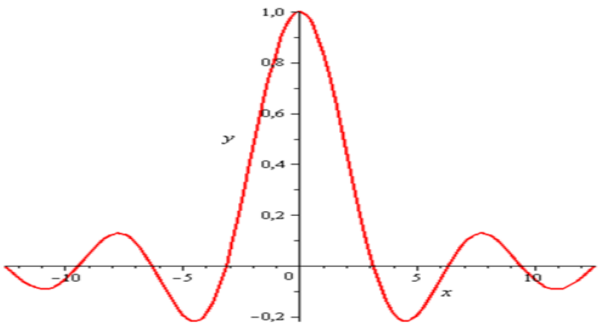
12)**font=[f,style,size]** – matnni chiqarish uchun shrift turini o`rnatish xususiyati: f shriftlar nomini bildiradi: TIMES, COURIER, HELVETICA, SYMBOL; style shrift stilini bildiradi: BOLD, ITALIC, UNDERLINE; size – pt da shrift katta kichikligini bildiradi.

13) **labels=[tx,ty]** – koordinata o'qlari yozuvini o'rnatish: tx – Ox o`qi bo`yicha va ty – Oy o`qi bo`yicha.

14)**discont =true** – cheksiz uzilishlarni yasash uchun ko`rsatmani bildiradi. plot buyrug`i yordamida y=f(x) funksiya grafigi bilan birga, ochiq ko`rinishda, parametrik berilgan y=y(t), x=x(t) funksiyalar grafigini ham yasash mumkin: plot([y=y(t), x=x(t), t=a..b], parameters).

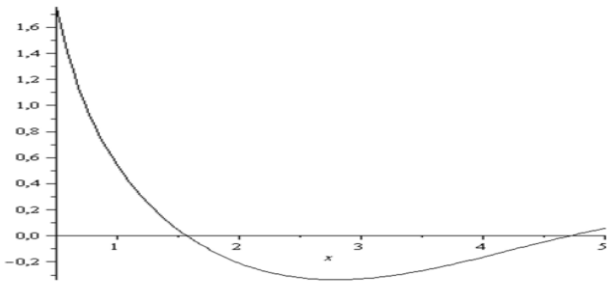
1. [-4π , 4π] intervalda y = sin x / x funksiya gafigini chizishni ko'rib chiqamiz. Buning uchun quyidagilar teriladi:

```
>plot(sin(x)/x,              x=-4*Pi..4*Pi,              labels=[x,y],
labelfont=[TIMES,ITALIC,12], thickness=2);
```



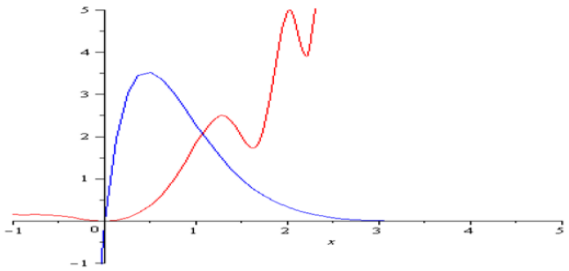
1.2.1-chizma. y = sin x / x funksiya grafigi.

```
> plot(cos(x)/x, x=0.5...5, color = black, numpoints = 100);
```



1.2.2-chizma.y=cos x / x funksiyasi grafigi.

```
>plot([x^2 + sin(x^3), 20 * exp(-2 * x) * sin(x)], x=-1..5,-1...5,
color = [red, blue], linestyle=1);
```



1.2.3-chizma.Yuqoridagi funksya grafigi.

1.3. Diagramma va gistogrammalar chizish asoslari

Diagramma vizualizatsiya metodlaridan foydalangan holda axborotning ramziy ifodasidir. Diagrammalar qadim zamonlardan beri qo'llanilgan, ammo mafkura davrida keng tarqalgan. So'zning grafigi ba'zida diagrammada sinonim sifatida ishlatiladi.

"Diagramma" atamasi keng tarqalgan ma'noda ishlatilishi mumkin, umumiy yoki o'ziga xos ma'noga ega:

- visual ma'lumotli qurilma: "illyustratsiya" atamasi kabi, "diagramma" barcha texnik janrlardan, jumladan grafikalar, texnik chizmalar va jadvallardan tashkil topgan bo'lishi mumkin.
- vizual ekranning o'ziga xos turi: bu chiziqlar, strelkalar yoki boshqa. havolalar bilan bog'langan shakllar bilan sifatli ma'lumotlarni ko'rsatadigan janrdir.

Fan falsafasi har ikki usulda ham qo'llaniladi. Misol qilib aytganda, "diagrammalar tasviriy, ammo mavhum, ma'lumotlarning namoyishi va xaritalar, chiziqli grafikalar, chizilgan grafikalar, muhandislik loyihalari va me'morlarning eskizlari diagrammalarning misolidir.

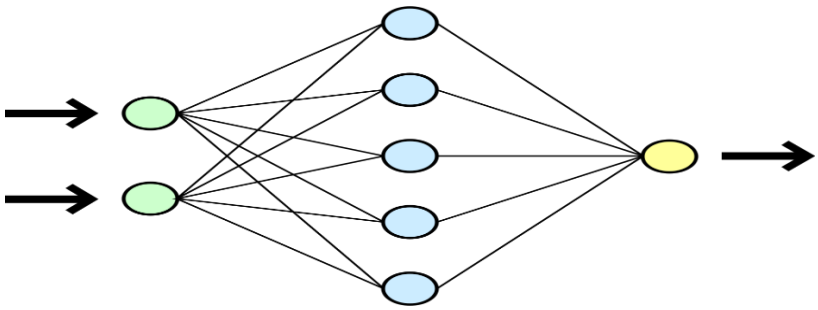
Aniq ma'noda grafikalar va kompyuter grafikalari, texnik illyustratsiyalar, infografika, xaritalar va texnik chizmalar bilan farq qiladi, "ma'lumotlarning tom ma'nodagi tasviri emas, mavhum. Diagrammaning mohiyati sifatida ko'rish mumkin formatlash qurilmalari shakli, nicel ma'lumotlar (raqamli ma'lumotlar) ko'rsatilmagan ekran emas balki aloqalar va mavhum ma'lumotlar,chiziqlar, o'qlar yoki boshqa havolalar bilan bog'langan geometrik shakllar kabilar kiradi.

Diagrammalar soddalashtirilgan raqamlar, karikaturalarning muhim ma'noni etkazish uchun mo'ljallangan.Ushbu soddalashtirilgan raqamlar odatda bir qator qoidalarga asoslanadi. Oq ga asosan asosiy shakli "nafislik, aniqlik, qulaylik, naqsh, soddaligi va haqiqiyligi" bilan ifodalanishi mumkin.

Asosiy diagramma turlari:

Grafikga o'xshash diagrammalar, ular orasida ma'lumotlar va munosabatlar to'plashni oladi va ularni har bir elementga 2D pozitsiyasini berib, ifodalar o'rtasida ma'lumotlar orasidagi o'zaro bog'liqlik sifatida ifodalanadi; Bunday texnikaning namunalari:

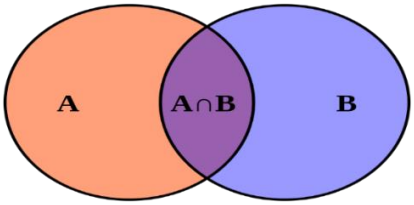
Daraxt strukturasi yoki daraxt diagrammasi grafik strukturadagi strukturaning ierarxik tabiatini ifodalashning bir usuli hisoblanadi. U "daraxt tuzilishi" deb ataladi, chunki klassik vakolat daraxtga o'xshaydi, hatto grafika aslida daraxt bilan qiyoslanadi, yuqorida "ildiz" va pastda "barglar" mavjud.



1.3.1-rasm.Daraxt diagrammasi

Grafika chizma matematika va kompyuter fanining geometrik grafik nazariyasi va axborot vizuallashtirish usullarini birlashtirib, ijtimoiy tarmoq

tahlillari, kartografiya, tilshunoslik va bioinformatika kabi ilovalardan kelib chiqadigan grafikalarning ikki o'lchovli tasvirlarini birlashtiradigan maydonidir.

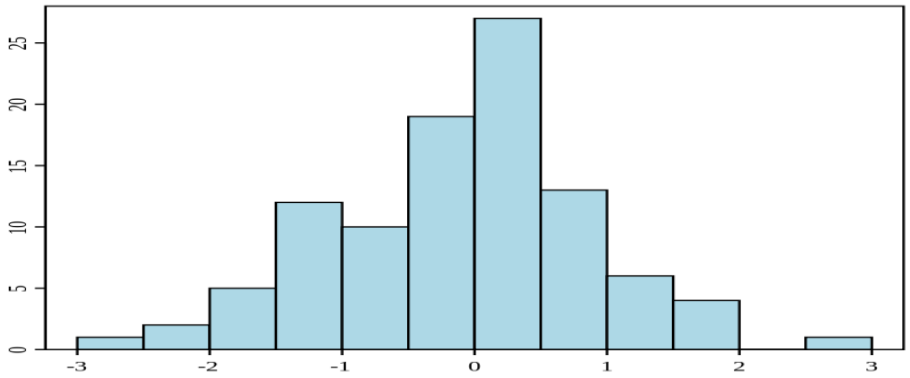


1,3.2-rasm.Venn diagrammasi

Venn diagrammasi - har xil to'plamlarning cheklanmagan to'plamlari o'rtasidagi barcha mumkin bo'lgan mantiqiy munosabatlarni ko'rsatuvchi diagramma. Ushbu diagrammalar elementlarni tekislikdagi nuqtalar sifatida tasvirlaydi va yopiq chiziqlar ichidagi maydonlar sifatida belgilanadi. Venn diagrammasi ko'pincha bir-biriga mos keladigan ko'plab chiziqli egri chiziqlar, odatda doiralar. S belgisi bilan belgilangan egri ichidagi fikrlar S to'plamining elementlarini ifodalaydi, chegaradan tashqaridagi fikrlar esa S belgisida bo'lmagan elementlarni ifodalaydi. Bu tasvirni oson o'qishni ta'minlaydi;

Masalan, S va T, S ∩ T ikkala jamoasining a'zolari bo'lgan barcha elementlar majmuyi S va T hududlarining bir-biriga kesishish maydoni bilan ifodalanadi.

Venn diagrammasi 1880 yilda Jon Venn tomonidan yaratilgan. Ular oddiy elementlar nazariyasini o'rgatish uchun ishlatiladi, shuningdek, ehtimollik,mantiq, statistika, tilshunoslik va kompyuter fanida sodda aloqalarni tasvirlaydi.



1.3.3-rasm.Raqamli ma’lumotlar tasvirlangan gitogramma.

Gistogramma raqamli ma'lumotlarning taqsimlanishining aniq ifodasidir. Gistogramma birinchi marotaba Karl Pearson tomonidan ishlab chiqilgan. Bir satr grafigi ikkita parametr bilan bog'liq degan ma'noni anglatadi, ammo gistogramma faqat bittasi bilan bog'liq. Gistogrammani yaratish uchun birinchi bosqich qadriyatlar diapazoniga, ya'ni butun qiymat oralig'ini bir qator ketma-ketliklarga bo'linib, keyin har bir oraliqda qancha qiymatning tushishini hisoblashdir. Ildizlar ko'pincha ketma-ket, o'zgaruvchan ketma-ketliklar sifatida aniqlanadi.

Gistogramalar ma'lumotlarning asosiy taqsimotining zichlik darajasini qo'zg'atadi va ko'pincha zichlikni baholash uchun pastki o'zgaruvchining ehtimollik zichligi funktsiyasini baholaydi. Imkoniyat zichligi uchun ishlatiladigan gistogrammaning umumiy maydoni har doim 1 ga normallashtiriladi. Agar x o'qi bo'yicha intervallarning uzunligi 1 bo'lsa, u holda gistogramma nisbatan chastota uchastkasi bilan bir xil bo'ladi.

Gistogramma javonlarning chastotalarini tekislash uchun yadrodan foydalanadigan sodda yadro zichligi hisob-kitoblari sifatida qaralishi mumkin. Bu esa, umuman olganda, asosiy o'zgaruvchining taqsimotini aks ettiradigan yanada aniq ehtimollik zichligi funksiyasini beradi. Zichlikdagi hisob-kitoblar gistogramma muqobil ravishda tuzilishi mumkin va odatda qutilar to'plami

o'rniga egri chiziladi. Gistogramlar ularning statistik xususiyatlarini modellashtirish zarur bo'lganda dasturlarda afzallik beriladi.

Gistogrammalar ba'zan chiziqli grafikalar bilan aralashtiriladi. Gistogramma uzluksiz ma'lumotlar uchun ishlatiladi, Ayrim mutaxasislar bar chartlaridagi farqlarni aniqlashtirish uchun to'rtburchaklar orasida bo'shliqlar mavjudligini tavsiya qiladi. Umuman, matematik ma'noda, histogram funksiyasi - bu har qanday ajratilmagan toifalarga kiradigan kuzatuvlar sonini hisobga oladigan funksiya. Bu histogramning grafigi faqat histogramni ifodalashning bir usuli hisoblanadi.

Ba'zi teoretiklar optimal sonini aniqlashga urinishgan, ammo bu uslublar odatda tarqatish shakli haqida kuchli taxminlar hosil qiladi. Haqiqiy ma'lumot taqsimotiga va tahlil maqsadlariga qarab, turli xil bin kengliklari mos kelishi mumkin, shuning uchun odatda mos kenglikni aniqlash uchun tajriba talab etiladi. Shu bilan birga, turli xil foydali qo'llanmalar va qoidalar mavjud.

Gistogramma uzluksiz ma'lumotlar to'plamining asosiy chastotasini taqsimlash (shakl) ni aniqlash va ko'rsatish imkonini beruvchi uchastkadir. Bu ma'lumotlar uning asosiy tarqalishi (masalan, an'anaviy taqsimot), chiqishi, skeletlari va boshqalar uchun tekshirilishiga imkon beradi. Gistogrammani va undagi xom ma'lumotlar quyidagicha ifodalanadi:

Gistogramlar maydonchaning balandligi emas, balki maydonga asoslangan.

Gistogrammada bu har bir quti uchun sodir bo'lgan chastotani ko'rsatadigan satr maydoni. Buning ma'nosi shuki, barning balandligi har bir binada qancha ballarni ko'rish mumkinligini ko'rsatmaydi.

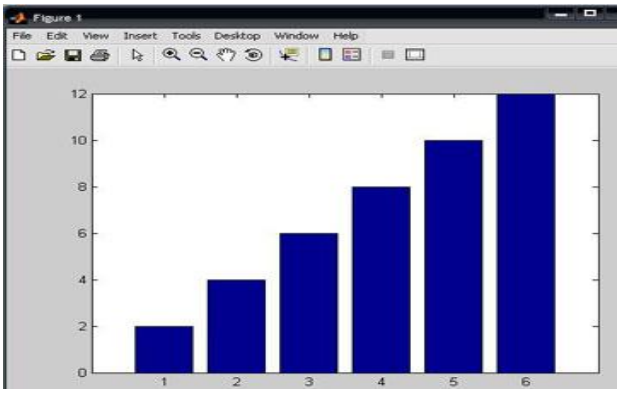
Grafika va gistogramma o'rtasidagi asosiy farq shundan iboratki, gistogramma faqat to'plamlarga bo'linib, to'plamlarga ajratilgan uzluksiz ma'lumotlar to'plamida ballar sonini tez-tez tekshirish uchun ishlatiladi. Boshqa

tomondan, chiziqli jadvallar oddiy va nominal ma'lumotlar silsilasini o'z ichiga olgan ko'plab boshqa turdagi o'zgaruvchilar uchun ishlatilishi mumkin.

Amaliy hisoblarda biror vektor tarkibini tasvirlaydigan ustunli diagrammalar deb ataluvchi gistogrammalar ko‘p uchraydi. Bunda vektorning har bir elementi balandligi uning qiymatiga mos bo‘lgan ustun shaklida ko‘rsatiladi. Ustunlar tartib raqamlariga va eng baland ustunning maksimal qiymatiga nisbatan ma’lum masshtabga ega bo‘ladi. Bunday grafiklar masalan, iqtisodiy o‘zgarish va boshqa jarayonlarni ifodalashi mumkin. Ular bar(a) komandasi yordamida yasaladi. Masalan:

```
>> a=[2 4 6 8 10 12];
>> bar(a)
```

Yuqoridagi komandani yozgan holda quyidagi gistogrammani tuzish mumkin:



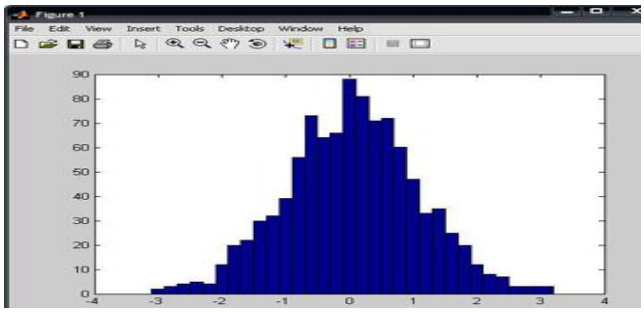
1.3.4-rasm.Gistogrammaga misol

Gistogramma tuzishning yana boshqa usuli ham mavjud bo‘lib, bu hist funksiyasi yordamida yasaladi:

- **N=hist(u)**- avtomatik tarzda tanlangan 10 intervalli vektor qiymatini qaytaradi;
- **N=hist(u,m)**- yuqoridagi kabi ishlaydi, faqat M (M-skalyar) intarvalda qaytaradi;

Quyidagi misolni ko‘rib chiqamiz:

```
>> x=-3:0.2:3; y=randn(1000,1);
>> hist(y,x); h=hist(y,x)
Columns 1 through 13
2     3     4     5     4    12    20    22    30    32    39    56
   73
Columns 14 through 26
64    66    88    81    71    72    60    47    33    35    25    20
    12
Columns 27 through 31
8     7     3     3     3
```



1.3.5-rasm.Yuqoridagi misolning gistogrammasi

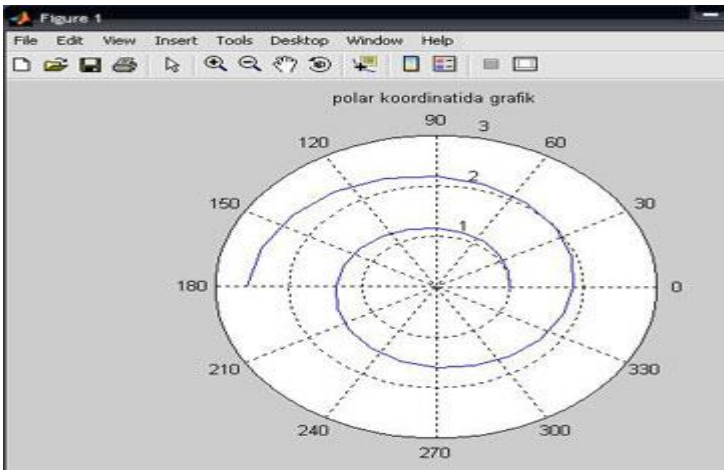
Qutbli koordinatalar tizimida ixtiyoriy nuqta joylashgan xuddi radius vektor oxiri kabi, koordinatalar tizimining boshlang‘ich nuqtasidan chiqib, RHO uzunlikka va THETA burchakka egaligini ko‘rsatadi. RHO(THETA) funksiya grafigini tuzish uchun quyidagi buyruqlardan foydalaniladi.THETA burchak odatda 0 dan 2π gacha o‘zgaradi. Qutbli koordinatalar tizimida funksiya grafigini tuzish uchun quyida keltirilgan buyruqlardan foydalaniladi :

- **polar(THETA,RHO)** - qutbli koordinatalar tizimida radius-vektor oxirining o‘z holatidagi RHO uzunlik bilan va THETA burchakni ko‘rsatib beruvchi grafikani tuzadi;
- **polar(THETA,RHO, S)** - analogli avvalgi buyruqda ishtirok etgan, lekin S qatorli konstanta yordamida tuzish uslulini analogli plot buyrug‘i asosida ruxsat beradi.

Quyidagi misolni ko’rib chiqamiz:

```
>> angle=0:.1*pi:3*pi;
```

```
>> r=exp(angle/10);
>> polar(angle,r),...
>> polar(angle,r);
>> title('polyar koordinatida grafik');
>> grid on
```



1.3.6-rasm.Yuqoridagi misolning grafigi.

I-BOB. JAVA DASTURLASH TILIDA INTERAKTIV MATEMATIK FUNKSIYALAR GRAFIGINI CHIZISH,DASTURIY MAJMUASINI ISHLAB CHIQISH

2.1. Java dasturlash tili

Java dasturlash tili - eng yaxshi dasturlash tillaridan biri bo'lib unda korporativ darajadagi mahsulotlarni(dasturlarni) yaratish imkoniyati mavjud. Bu dasturlash tili Oak dasturlash tili asosida paydo bo'ldi. Oak(ma'nosi eman daraxti) dasturlash tili 90-yillarning boshida Sun Microsystems(hozirda Oracle nomidan ish yuritadi) tomonidan platformaga(operatsion tizimga) bog'liq bo'lmagan holda ishlovchi yangi avlod aqlli qurilmalarini yaratishni maqsad qilib harakat boshlagan edi. Bunga erishish uchun Sun hodimlari C++ ni ishlatishni rejalashtirishgan edi, lekin ba'zi sabablarga ko'ra bu fikridan voz kechishdi. Oak muvofaqiyatsiz chiqdi va 1995- yilda Sun uning nomini Java ga almashtirdi, va uni WWW rivojlanishiga hizmat qilishi uchun ma'lum o'zgarishlar qilishdi.

Java dasturlash tili 1990 yillarda ishlab chiqarila boshlangan bo'lsa ham, uning birinchi versiyasi(Java 1.0) 1996 yil ommaga taqdim etilgan. Undan so'ng keyingi versiyalar sekin-astalik bilan chiqa boshladi: 1998 yil - Java 2, 2004 yil - Java 5.0, 2006 yil - Java 6, 2011 yil - Java 7, 2014 yil - Java 8. Java Obyektga yo'naltirilgan dasturlash tili va u C++ ga ancha o'xshash. Eng ko'p yo'l qo'yildigan xatolarga sabab bo'luvchi qismalari olib tashlanib, Java dasturlash tili ancha sodda tarzda ishlab chiqildi. Java texnologiyasi o'ta sodda, xavfsizlikni yuqori darajada ta'minlab bera oladigan, kuchli, to'la obyektga yo'naltirilgan dasturlash tili bo'lib, muhit (platforma)ga bog’liq bo'lmagan holda ishlaydi. U bilan xatto eng kichik qurilmalarga ham dasturlar yozish imkoniyati mavjud.

Java texnologiyasi to'laligicha Java Virtual Machine(JVM) ga asoslangan. Java Virtual Machinening vazifasi tarjimonlik ya'ni, dastlab biz yozgan *.java fayl kompilyator yordamida bayt kodga o'giriladi va Java Virtual Machine

yordamida esa mashina tiliga aylantiriladi. Bundan ko’rinib turibdiki Java Virtual Machine qaysi platformaga tegishli bo'lsa, kodlarni ham o'sha platformaga moslab beradi.

Java dasturlash tilining imkoniyatlari:

- ✓ WORA - Write Once, Run Anywhere (portable). Platforma tanlamaydi;
- ✓ Xavfsizlik (ishonch yo’q kodni havfsiz ishga tushirish);
- ✓ Xotirani xavfsiz boshqarish (avtomat ravishda musorlarni yig’adi);
- ✓ Tarmoq uchun dasturlar yozish ;
- ✓ Ko’p oqimli (Multi-thread) dasturlash;
- ✓ Dinamik kengaytirish;
- ✓ klasslar alohida fayllarda saqlanadi. Kerak bo’lsa ishlatiladi. Agar kerak bo’ls dinamik ravishda imkoniyatini oshirish ham mumkin.

JAVA tilining asosi bu - ob'ektga yo'naltirilgan dasturlash (OYD) ning prinsiplarini amalga oshirish. Ob'ektga yo'naltirilgan metodika JAVA dan ajiralmas va barcha JAVA ning dasturlari qandaydir bir ob'ekt yo'nalishiga egadir. Shuning uchun oddiy JAVA dasturini yozishdan oldin, OYD haqida tushunchaga ega bo'lishimiz kerak. OYD - bu kuchli vosita bo'lib dasturlashni takomillashtiruvchi prosess. Kompyuter kashf etilgandan boshlab murakkab dasturlarni soddalashtirish maqsadi dasturlash usullari ko’p marotaba o'zgardi. Masalan birinchi kompyuterlar bilan dasturlashtirish mashinaning qo'llanish qoidalari yordami bilan amalga oshirilgan yoki bu usul dastur uzunligida instruksiyalar bir nechta bo'lganiga qadar ishladi. Dasturning hajmi o'sishiga boqliq assembler tili yaratildi. Bu til dasturchilarga mashina instruksiyasining simvolik ko'rinishidan foydalanib yanada bundan kattaroq va qiyiniroq dasturlarni yozish imkoniyatini berdi. Dasturning hajmining o'sishiga boqliq turda yuqori darajadagi tillar (masalan, FORTRAN , COBOL) paydo bo'ldi va dasturchilarga o'sib borayotgan dastur qiynchiliklarni engishga yordam berdi. Bu

birinchi kompyuter tillari kritik holatga yaqinlashganda strukturali dasturlash paydo bo'ldi.

Ob'ektga yo'naltirilgan dasturlash umuman boshqacha ishlaydi. Ular ma'lumotlar atrofida tashkil qilingan bo'lib,bunday tashkil qilishning kalitlik prinsipi: ma'lumotlargina kodga kirish mumkinligini boshqaradi deyiladi. Obektga yo'naltirilgan tillarda programmist ma'lumotlar ustida harakatlarning bajarilishi uchun ruhsat beradigan ma'lumot va kodni aniqlaydi. Shunday qilib, ma'lumotlar turi unga qo'llanadigan operasiyalarni aniq aniqlaydi. Ob'ektga yo'naltirilgan dasturlashni qullab quvvatlash uchun barcha OYD tillari shuningdek JAVA ham uch turdagi harakterlik belgilardan iborat:

- ✓ **Inkapsulyasiya**
- ✓ **Polimorfizm**
- ✓ **Meros.**

Inkapsulyasiya

Inkapsulyasiya - bu kod(harakat) va ma'lumotlarni birlashtiradigan va noto'qri qullanishdan saqlaydigan dasturlash mehanizmi. Ob'ektga yo'naltirilgan tillarda kod va ma'lumotlarni shunday qilib bog'lasa bo'ladi, bunda avtanom qora quti paydo bo'ladi. Bu qutining ichida barcha zarurli bo'lgan ma'lumotlar va kod joylashgan bo'ladi. Kod va ma'lumotlarni bunday bog'lashdan ob'ekt qosil bo'ladi. Boshqacha aytganda ob'ekt -- bu inkapsulyasiyani qullab quvvatlaydigan element.

Kod va ma'lumotlar bo'lar ikkalasi obekt tuzuvchilari bo'lib, ular ichkaridan yopilgan va ochiq bo'lishi mumkin. Yopiq kod yoki yopiq ma'lumotlar ob'ektning boshqa qismlarigagina belgili va ruhsat etilgan. Bu yopiq kod yoki ma'lumotlar shu ob'ekt ichida bo'lmagan dasturning boshqa qismlariga hech qanday aloqa qilishi mumkin emasligini anglatadi. Agar kod yoki ma'lumotlar ochiq bo'lsa, u holda u dasturning boshqa qismlari(ularning ob'ekt ichida bo'lganligiga qaramasdan) ham aloqa qila olish mumkinchiligiga ega

bo’ladi. qoida bo'yicha ob'ektning ochiq qismi yopiq elementlar bilan boshkariluvchi interfeysni taminlash uchun foydalaniladi.

JAVA inkapsulyasiyasiningsh asosiy birligi sinf xisoblanadi.Sinf ob'ekt shaklini aniqlaydi. U kod va ma'lumotlar singari berilgan ma'lumotlarni tuzatib boradi. JAVA da sinf ob'ektlarni tuzish uchun foydalaniladi. Ob'ekt - sinflarningning nusxasi. Shunday qilib sinf ob'ektni qanday qurishni ko'rsatuvchi shablondagi elementlar yig’indisi.

Sinfni tuzuvchi kod va ma'lumotlar sinf a'zolari deb nomlanadi. Sinfda aniqlangan ma'lumotlar nusqa o'zgaruvchilari (instanse variable) deb ataladi, shu ma'lumotlar bilan ishlovchi kod esa a'zo metodlari(member method) yoki soddagina qilib metodlar deb aytiladi. «Metod» -bu termin bo'lib, JAVA uchun qism dasturlani anglatish uchun qabo’l qilingan.

O'z navbatida JAVA C++ning avlodi, “funksiya” termini ham JAVA-metodi singari qabo’l qilingan bo'lib foydalaniladi.

Polimorfizm

Polimorfizm (grekcha polymorphism so'zdan olingan bo'lib, «ko'p form»degan ma'noni anglatadi) - bu butun sinf uchun bir interfeysni yaratuvchi harakat sifati. Polimorfizmga avtomobil ruli oddiy misol bo'la oladi. Rul (interfeys) bu avtomobilda qanday rul mehanizmini foydalanishga qaramasdan rul bo’lib qoladi. Boshqacha aytganda har qanday holatda bir hil ishlaydi: sizning avtomobilni qanday boshqarganingiz bilan ham u ruldir. Shunday qilib qanday rullik boshqarish bo'lishidan qat'iy nazar rul chapga burilsa u avtomobilni m chapga burishga majbur qiladi. Bir obrazli interfeys imkoniyati shundan turadiki, ya'ni agar siz rul bilan qanday ishlashni bilsangiz, unda siz xoxlagan turdagi avtomobilni boshqara olasiz.

Dasturlashga ham shunday prinsipni qo'llanishga bo'ladi. Misol sifatida uchun “ohirgi kelgan-birinchi hizmatda” prinsipi bo'yicha funsiyalanuvchi stek (stack), shuningdek Xotira oblastini qarab o'tamiz.Misol uchun Uch hil turdagi stekni tashkil qilish kerak bo'lgan dastur yozasiz. Birinchi stek -butun sonli

qiymat uchun, ikkinchisi qalqib yuruvchi nuqta(tochka) qiymati uchun,uchunchisi simvollar uchun. Bunday holda har bir stekni amalga oshirish uchun har hil turdagi saqlangan ma'lumotlar bo’lishiga qaramasdan bir hil turdagi algoritm foydalaniladi. Obektga yunaltirilgan til bo’lmagan holda sizga uchta har hil nomlarga ega bo'lgan “steklar” qism dasturining tuplamini tuzishingizga tuqri kelardi. Biroq polimorfizm bo’lganligi sababli JAVA muqitida barcha stekning uch turini ham o'z ichiga oladigan “steklar” qism dasturining bir tuplamini tuzish etarli. Boshqacha aytganda bir stekni qo'llanishni bilgan holda boshqalari uchun ham qo'llanishga bo'ladi.

Polimorfizm konsepsiyasi ko'pincha quyidagi so'zlar bilan ifodalanadi:

«bir interfeys - ko'p metodlar». Bu belgili harakatlar gruqi bajarilishi uchun umumiy interfeys ishlab chiqish mumkinligini bildiradi. Polimorfizm dasturning qiyinchiligini osonlashtirishga yordam beradi bu erda dasturchiga umumiy sinf harakati vazifasi uchun bitta interfeysdan foydalanish imkoniyatini beradi. Konkret (shuningdek karakli yoki teskari holatda) harakatni (metod) kompilyator tanlaydi. Dasturchiga buni o'zi qilishiga zarurat bo’lmaydi. Uning vazifasi - umumiy interfeysni to'qri foydalanish.

Meros

Meros bu shunday jarayon bo'lib unda bir ob'ekt ikkinchi birining hossasiga utishi mumkin. Meros sababli ierarhiyalik klassifikasiya qo'llab quvvatlanadi. Boshqariluvchi ierarhicheskoy (kelib chiqqan) klassifikasiya ko'rinishida katta bo'lgan bilimlar soqasi tashkil qilinadi. Masalan, Krasniy Delishes olmasi olma klassifikasiyasining qismi xisoblanadi, o'z navbatida mevalar sinfining ham qismidir va shuningdek katta bo'lgan ovqat sinfining ham qismi xisoblanadi. Shunday qilib ovqat sinfi mevalar qismsinfiga(podklass) ham tegishli bo’lgan belgili bir sifatga ega (eb bo'ladigan,ishtaqalik va b .) bo'ladi. Bu sifatlardan mevalar sinfi meva maqsulotlarini boshqalaridan ajratib turadigan o'ziga tan bo'lgan jiqatlarga (spesifikalik harakter) (shirinlik, shirali va b.)egadir Olma sinfida esa olmaga tan bo'lgan sifatlar aniqlanadi(terakda o'sadi tropik emas

va b.). Krasniy Delishes sinfi dastlabki barcha sinflarni meros qilib olgan holda olmaning bu naviga tegishli bo’lgan sifatni aniqlaydi.

Agar belgilarning ierarhik(kelib chiqish) berilishini xisobga olmasak, u holda har bir ob'ekt uchun tan bo’lgan barcha uning harakteristikasining aniq shaklini aniqlashga to'qri kelardi. Lekin obekt merosiga boqliq uning sinf ichida a'lohidaligini bildiruvchi sifatlarinigina aniqlash etarlidir, chunki u(ob'ekt) o'z ota-onasining umumiy belgilarini meros qiladi. O'z navbatida meros mehanizmigina bir ob'ekt umumiy sinf uchun konkret ekzemplyarligini beradi.

O'zgaruvchilar va konstantalar

O'zgaruvchilar. O'zgaruvchi nomi ostiga chizish belgisi yoki lotin harfidan boshlanuvchi lotin harflari, va ostiga chizish hamda dollar belgilari ketma ketligi ya'ni identifikatordir.

Java - tilida registr farq qiladi. Masalan, Value va VALUE - turli identifikatorlardir.

O'zgaruvchilarning quyidagi tiplari mavjuddir: byte(bayt), char(simvol), short(qisqa butun), int(butun), long(uzun butun), float(haqiqiy), double(ikkilangan haqiqiy), boolean(mantiqiy).

JAVA tilida unsigned (ishorasiz) ta'rifi ishlatilmaydi.

O'zgaruvchilar ta'rifi sodda shakli:

<tip> <o'zgaruvchilar_nomlari_ro'yhati >; o'zgaruvchilarni ta'riflashda boshlang’ich qiymatlarini ko'rsatish mumkin.

<tip> <o'zgaruvchilar_nomlari_ro'yhati >= <inisializator>;

Bu usul inisializasiya deyiladi.

Misollar:

```
float pi = 3.14 , cc=1.3456;
int year = 1999;
```

Agar o'zgarvchi qiymati ko'rsatilmagan bo'lsa avtomatik ravishda noll qiymat qabo’l qiladi.

Konstantalar. Konstanta bu o'zgartirish mumkin bo'lmagan qiymatdir. Konstantalar butun, simvolli haqiqiy, mantiqiy turlari mavjud.

Butun sonlar o'nlik, sakkizlik yoki o'n oltilik sanoq sistemalarida berilishi mumkin. O'nlik konstantalar o'nlik raqamlari ketma ketligidan iborat bo'lib, birinchi raqami 0 bo'lishi kerak emas(masalan: 8, 0, 192345). Sakkizlik konstantalar 0 bilan boshlanuvchi sakkizlik raqamlaridan iborat ketma ketlikdir (masalan: 016 - o'nlik qiymati 14, 01). O'n oltilik konstantalar - 0x yoki 0X bilan boshlanuvchi o'n oltilik raqamlar ketma ketligidir (masalan: 0xA, 0X00F).

Simvolli konstanta - apostrifga olingan bitta (masalan: 'q','r,'6') yoki bir nechta (masalan: -'\n', '\0hF5') simvol. Slesh '\' simvolidan boshlangan simvollar eskeyp yoki boshqaruvchi simvollar deyiladi.

Haqiqiy konstanta ikki ko'rinishda bo'lishi mumkin: fiksirlangan nuqtali (masalan: 5.7, .0001, 41.) va suzuvchi nuqtali (masalan: 0.5e5, .11e-5).

Mantiqiy konstantalar true(rost) va false(yolqon) qiymatlardan iborat. Ichki ko'rinishi false - 0, ihtiyoriy boshqa qiymat true deb qaraladi.

Satrli konstantalar Java tilida juft qavslar ichiga olingan ihtiyoriy matndir (""). Satrli konstantalar bir qatorga joylashishi lozim.

Amallar

Arifmetik amallar. Arifmetik amallar binar va unar amallarga ajratiladi. Binar amallarga qo'shish +, ayirish -, ko'paytirish *, bo'lish / va modul olish % amallari kiradi.

Masalan 20/3=6; (-20)/3=-6; 5%2=1;

Unar amallarga unar minus - va unar +; inkrement ++ va dekrement- amallari kiradi. Inkrement va dekrement amallari prefiks ya'ni ++i, postfiks ya'ni i++ ko'rinishda ishlatishi mumkin. Masalan agar i=2,k=2 bo'lsa u holda 3+(++i)= 6, 3+k++=5 ga teng bo'ladi. Ikkala holda ham i=3,k=3 ga teng bo'ladi.

Nisbat amallari. Nisbat amallari qiymati mantiqiy bo'lib katta >; kichik <;katta yoki teng >=; kichik yoki teng <=; teng == ; teng emas != amallaridan iborat.

Mantiqiy amallar. Mantiqiy amallar diz'yunksiya ||, kon'yunksiya &&, inkor ! amallaridan iborat.

Razryadli amallar. Razryadli amallar Razryadli diz'yunksiya |, Razryadli kon'yunksiya &, Razryadli XOR ^, Razryadli inkor !, Razryadli chapga surish<< va Razryadli o'ngga surish >> amallaridan iborat. Masalan 5 kodi 101 ga teng va 6 kodi 110 ga teng:

```
 6&5= 4=100; 6|5= 7= 111; 6^5=3=011; ~6=4=010.
5<<2=20 yoki 101<<2=10100;  5>>2=1 yoki 101>>2=001=1.
```

Qiymat berish amali. Oddiy qiymat berish amali binar amal bo’lib chap operandi odatda uzgaruvchi ung operandi odatda ifodaga teng bo’ladi:

O'zgaruvchi_nomi = ifoda;

Masalan: Z=4.7+3.34; C=y=f=4.2+2.8;

Murakkab qiymat berish amali unar amal bo'lib quyidagi ko'rinishga ega:

O'zgaruvchi_nomi amal= ifoda;

Masalan: x+=4 ifoda x=x+4 ifodaga ekvivalentdir;

x>>=4 ifoda x=x>>4 ifodaga ekvivalentdir;

Shartli amal. Shartli amal ternar amal deyiladi va uchta operanddan iborat bo’ladi: <1-ifoda>q<2-ifoda>:<3-ifoda>. Masalan: y=a<bqa:b .

Tiplar bilan ishlovchi amallar. Tiplarni o'zgartirish amali quyidagi ikki kurinishga ega:

Kanonik: (tip_nomi) operand; masalan: r=(unsigned long)1;

Funksional: tip_nomi (operand); masalan: z=double(1);

Xotiradagi hajmni hisoblash sizeof amalining ikki ko'rinishi mavjud:

sizeof ifoda masalan: Sizeof 3.14=8 sizeof (tip) masalan: Sizeof(char)=1

Ko'rsatkichlar va ilovalar

Ko'rsatkichlar ta'rifi. Ko'rsatkichlar qiymati konkret tipdagi ob'ektlar uchun Xotirada ajratilgan adreslarga tengdir.

Biror o'zgaruvchi adresini olib, ko'rsatkichga qiymat sifatida berish uchun adres olish «&» amalidan foydalaniladi. Ko'rsatkich orqali o'zgaruvchi qiymatini olish uchun adres bo'yicha qiymat olish «*» amalidan foydalaniladi.
Ko'rsatkichlarga o'hshab Ilovalarning qiymatlari ham adreslardir. Ilovalarga murojaat qilinganda avtomatik ravishda * qiymat olish amali bajariladi.

Dastur tuzilishi

Java tilida dastur nomlangan sinflardan iborat.

Misol:

```
class HelloWorld {
public static void main (String args []) {
System.out. println ("Hello World");
}
}
```

Yuqorida keltirilgan misol matnini HelloWorld.java faylga yozish lozim.

Bu misolni translyasiya qilish uchun javac - Java translyatorini ishga tushirish lozim:

```
C: \> javac HelloWorld.Java
```

Translyator HelloWorld.class nomli prosessorga boqliq bo'lmagan bayt-kodli fayl yaratadi. Dastur kodini ishga tushirish uchun Java tili muhitiga bajarish uchun yangi sinfni yuklash lozim. Shuni ta'kidlash lozimki sinf joylashgan fayl emas sinf nomi ko'rsatiladi.

```
C:\> java HelloWorld
Hello World
```

Murojaat modifikatori public asosiy usul main hamma sinflarda ko'rinishini bildiradi.

Maxsus static so'zi yordamida butun sinf bilan birga ishlatiladigan o'zgruvchilar va usullar ta'riflanadi. Agar usul static so'zi yordamida ta'riflangan bo'lsa faqat lokal va statik o'zgaruvchilar bilan ishlashi mumkin.

Agar main usuli qiymat qaytarmasa modifikator void ishlatiladi.
Hamma mavjud Java-interpretatorlar, sinf interpretasiyasini main usulini chaqirishdan boshlaydi. Java-translyator main usuli bo'lmagan sinfni translyasiya

qilishi mumkin, lekin Java-interpretator main usuli bo'lmagan sinfni ishga tushira olmaydi.

Usul String args[] elementi String sinfiga tegishli ob'ektlar massivi bo'lgan args parametrni e'lon qiladi .

Ekranga chiqarish uchun out ob'ektining println usulidan foydalaniladi. Ob'ekt out statik ravishda System inisializasiya qilinib OutputStream sinfida ta'riflangandir.

Nomlangan konstantalar

Java tilida konstantalarni belgilash uchun final kalit so'zidan foydalaniladi:

```
public class Constants
public static void main(String[] args)
final double CM_PER_INCH = 2.54;
double paperWidth = 8.5;
double PaperHeight = 11;
System.out.println("Sahifa hajmi santimetrlarda: "
+ paperWidth * CM_PER_INCH + "ga"
+ paperheight * CM_PER_INCH);
```

Nomlangan konstanta qiymatini o'zgartirish mumkin emas. Konstantalar nomida katta harflarni ishlatish shart emas.

Java tilida bir sinf ichida bir necha usullarda ishlatiladigan konstantalar sinf konstantalari (class constants) deb ataladi. Sinf konstantalari static final kalit so'zi yordamida ta'riflanadi.

```
public class Constants2
{
public static final double CM_PER_INCH = 2.54;
public static void main(String [] args)
{
double paperWidth = 8.5;
double PaperHeight = 11;
System.out.printIn("Sahifa kattaligi santimetrlarda: "
+ paperWidth * CM_PER_INCH + "ga"
+ paperHeight * CM_PER_INCH);
```

Sinf konstantalari main usuli tashqarisida ta'riflanadi. A8gar konstanta. Agar konstanta public sifatida ta'riflangan bo'lsa unga boshqa sinf usullarida murojaat qilish mumkin. Masalan quyidagicha:

```
Constants2.CM_PER_INCH.
```

Java tilida const so'zi hizmatchi hisoblanadi, lekin foydalanilmaydi. Konstantalarni ta'riflash uchun final kalit so'zidan foydalanish zarur

Tiplarni keltirish

Tiplarni keltirish (type casting) ma'lum tipdagi o'zgaruvchi boshqa tidagi qiymat qabo’l qilganda foydalaniladi. Ba'zi tiplar uchun keltirish avtomatik ravishda bajariladi. V Java tilida avtomatik tiplarni keltirish o'zgaruvchi tipi hajmi qiymatni saqlashga etarli bo'lganda bajariladi. Bu jarayon kengaytirish (widening) yoki yuksaltirish (promotion) deb ataladi, chunki, kichik razryadli tip katta razryadli tipga kengaytiriladi. Masalan int tipi byte tipidagi qiymatni saqlashga etarli, shuning uchun tiplarni keltirish talab qilinmaydi. Teskarisi mumkin emas, shuning uchun byte tipidagi o'zgaruvchi int tipidagi qiymatni qabo’l qilishi uchun, tiplarni keltirish operatoridan foydalanish lozim. Bu jarayon toraytirish (narrowing) deb ataladi, chunki translyatorga qiymatni o'zgartirish haqida oshkor ma'lumot beriladi. Buning uchun dumaloq qavs ichida tip nomi ko'rsatiladi. Masalan:

```
int a = 100;
byte b = (byte) a;
```

Agar suzuvchi qavsli sonni eng yaqin butun songa keltirish lozim bo'lsa Math.round usulidan foydalaniladi.

```
double x = 9.997;
int nx = (int)Math.round(x) ;
```

Endi nx o'zgaruvchi qiymati 10 ga teng. Lekin round usulidan foydalanilgan tiplarni keltirish operatoridan foydalanish lozim chunki bu usul long tipidagi qiymatni qaytaradi.

Sonni bir tipdan ikkinchisiga keltirishda natija zarur diapazondan chiqib ketishi mumkin, bu holda natija qisqartiriladi. Masalan (byte) 300 ifoda qiymati 44 ga teng.

Mantiqiy va butun tiplar orasida keltirish mumkin. Ba'zi hollarda mantiqiy qiymatni butun qiymatga keltirish uchun shartli ifodadan foydalanish mumkin, masalan b q 1 : 0.

Ifodalarda tiplarni avtomatik keltirish

Agar ifodada byte, short va int tipidagi o'zgaruvchilar ishlatilsa, butun ifoda tipi int ga ko'tariladi. Agar ifodada biror o'zgaruvchi tipi- long bo'lsa, butun ifoda tipi long tipga ko'tariladi. Ko'zda tutilgan bo'yicha Java tilida hamma butun konstantalar int tipiga ega deb qaraladi. Hamma butun konstantalar ohirida L yoki 1 simvoli turgan bo'lsa, long tipiga ega.

Agar ifoda float tipidagi operandga ega bo'lsa, butun ifoda float tipiga ko'tariladi. Agar biror operand double tipiga ega bo'lsa, butun ifoda tipi double tipiga ko'tariladi. Ko'zda tutilgan bo'yicha Java tilida hamma suzuvchi vergulli konstantalar double tipiga tegishli hisoblanadi. Agar haqiqiy konstanta ohirida F yoki f simvoli turgan bo'lsa, float tipiga ega.

2.2. JavaFX bilan ishlash

JavaFX - har xil qurilmalar bo'yicha ishlaydigan ish stoli ilovalarini yaratish va etkazib berish uchun mo'ljallangan dastur hisoblanadi.

JavaFX Microsoft Windows, Linux va macOS kompyuterlar va veb-brauzerlarni qo'llab-quvvatlaydi. JavaFX endi eng so'nggi Java bilan birga to'planmaydi yoki Oracle tomonidan qo'llab-quvvatlanmaydi, ayni paytda u hali ham uzoq muddatli Java SE 8-dan 2022 yilgacha qo'llab-quvvatlanadi.

JavaFX 2.0 versiyasidan oldin, ishlab chiquvchilar JavaFX ilovalarini yaratish uchun JavaFX skript deb nomlangan, statik usulda yozilgan deklarativ tildan foydalanganlar. JavaFX skriptlari Java bytecode uchun tuzilganligi sababli, dasturchilar Java kodini ham ishlatishlari mumkin. JavaFX ilovalari Java SE-ni yoki Java ME-ni ishga tushiradigan har qanday uyali telefonda ishlaydigan har qanday ish stoli ustida ishlashi mumkin.

JavaFX 2.0 va undan keyingi versiyalari "native" Java-kutubxonasi sifatida qo'llaniladi va JavaFX-ni ishlatadigan dasturlar "native" Java kodida yoziladi.

JavaFX skriptlari Oracle tomonidan olib tashlangan, ammo Visage loyihasida rivojlanish davom etmoqda.JavaFX 2.x, Solaris operatsion tizimi yoki mobil telefonlarini qo'llab-quvvatlamaydi.

JavaFX ish stollarida Windows Vista, Windows 7, Windows 8, Windows 10, macOS va Linux operatsion tizimlarini qo'llab-quvvatlaydi. JavaFX Mobile 1.x uyali telefonida Symbian OS, Windows Mobile va xususiy real vaqtda operatsion tizimlar kabi bir nechta mobil operatsion tizimlarda ishlashga qodir.

Ochiq kodli JavaFXPorts iOS (iPhone va iPad), Android va o'rnatilgan (Raspberry PI); ”Gluon" nomi bilan yaratilgan savdo dasturlari qo'shimcha funktsiyalar va ish stoli bilan bir xil mobil platformalarni qo'llab-quvvatlaydi.Bu ish stoli, iOS va Android qurilmalari uchun ilovalar yaratish uchun yagona manba kodi bazasini yaratadi.

Java texnologiyalari Java SE (Java Standart Edition) - serverda, shaxsiy kompyuterda desktoplarda ishlovchi dasturlar, appletlar yaratish uchun foydalaniladi. Bu texnologiya yordamida yaratilgan dasturlar deyarli barcha operatsion tizimlarda ishlay oladi(Windows NT, Macintosh,Linux va Solaris). Shu bilan birga JavaSE boshqa Java turlarining asosi hisoblanadi. Java EE (Java Enterprise Edition) - Java texnologiyalari orasida eng keng tarqalgan turi xisoblanib unda asosan serverda ishlovchi dasturlar yaratiladi, masalan ko'p foydalanuvchili web-saytlar yaratishda keng qo'llaniladi va asosan internetda ishlovchi dasturlarda ishlatiladi.

Java SE ni Java EE dan eng asosiy farqi Java EE o'z tarkibiga Java SE ni olibgina qolmay shu bilan birga ko'pgina boshqa qo'shimcha kutubxonalarni(odatda *.jar) ham o'z ichiga oladi ya'ni: Servlet, JavaMail, JSF(Java Server Face) va boshqa ko'pgina internetga asoslangan qoshimcha kutubxonalar. Java ME (Java Micro Edition) - Java SE ning ba'zi qismlarini o'z

ichiga oladi, JavaME yordamida kichik qurilmalar uchun dastrular yozish mumkin, masalan, mobil telefon uchun o'yinlar, dasturlar yaratish mumkin.

Javada kompilyator biz yozgan kodni bayt-kodga o'giradi, odatda kompilatsiyadan o'tgan klasslar *.class qisqartirmasi bilan tugaydi va kompilatsiyadan o'tgan klassni Java Virtual Machine(JVM) ga yuklanadi va bayt-kodli fayllarni interpretatsiya qiladi, ya'ni mashina tiliga o'giradi va shu bilan birga undagi kodni imkoni boricha optimallashtiradi. Java dasturlash tilida dastur tuzish uchun, dastlab, kompyuterga kerakli dasturlarni o‘rnatish lozim. Birinchidan, Java dasturlarni ishga tushirish uchun, bizga Java-mashina kerak bo‘ladi. Gap shundaki, barcha Java dasturlar faqat Java- mashina o‘rnatilgan kompyuterlarda ishlaydi. Java-mashina Java dastur uchun muhit hisoblanadi. Ikkinchidan, Java dasturlarni yozish uchun maxsus muhit(IDE) kerak bo‘ladi. Ikkala dasturni ham oraclening saytidan yuklab olsa bo’ladi. Yuklab olish uchun Oracle saytiga kiriladi va Java Platform(JDK)ni yuklab olinadi.

JDKni yuklab oladigan oynaga o‘tiladi va litsenziyaga rozi bo‘lilsh (Accept License Agreement) tugmasi bosiladi. Keyin kompyuterga mos keladigan JDKni tanlanadi, agar 32 bitlik tizim bo‘lsa, "jdk-8u20-windows-i586.exe"ni tanlanadi va yuklash boshlash tugmasi bosiladi

Product / File Description	File Size	Download
Linux x86	135.24 MB	jdk-8u20-linux-i586.rpm
Linux x86	154.87 MB	jdk-8u20-linux-i586.tar.gz
Linux x64	135.6 MB	jdk-8u20-linux-x64.rpm
Linux x64	153.42 MB	jdk-8u20-linux-x64.tar.gz
Mac OS X x64	209.11 MB	jdk-8u20-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	137.02 MB	jdk-8u20-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	97.09 MB	jdk-8u20-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	137.16 MB	jdk-8u20-solaris-x64.tar.Z
Solaris x64	94.22 MB	jdk-8u20-solaris-x64.tar.gz
Windows x86	161.08 MB	jdk-8u20-windows-i586.exe
Windows x64	173.08 MB	jdk-8u20-windows-x64.exe

Yuklab olingandan so‘ng, dasturni ishga tushiriladi va berilgan savollarga qarab o‘rnatiladi(unchalik qiyiin ish emas va ko‘p vaqt ham kerak emas).

Odatda JDK "C:\Program files\Java" adresiga o‘rnatiladi.

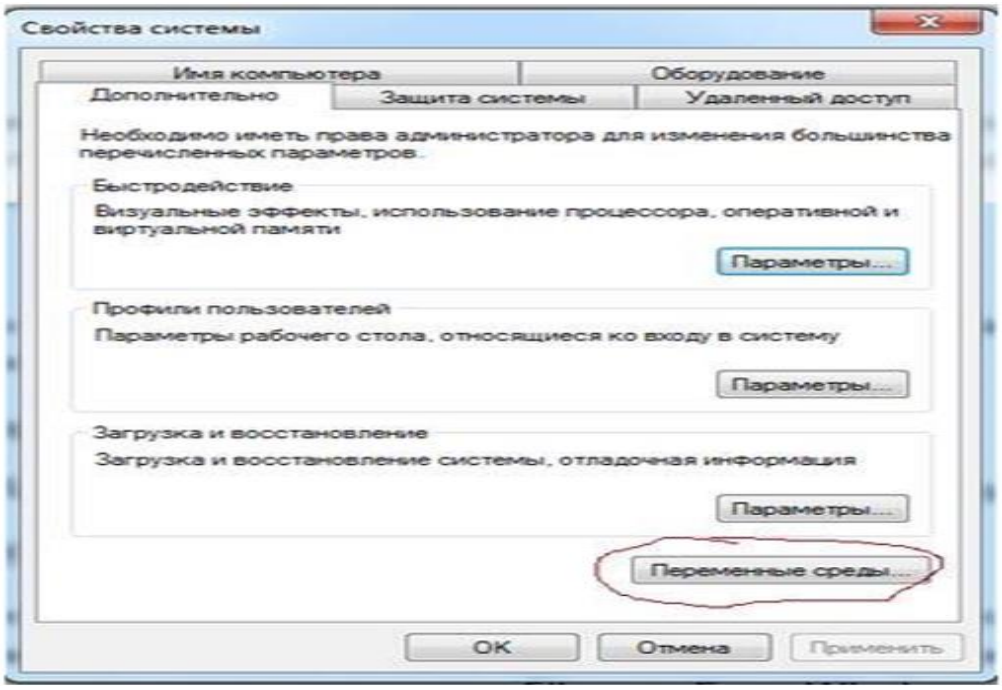
Keyingi bosqichda, java bajaruvchi utilitlarini topamiz. Uning uchun quyidagi papkaga kiriladi:

C:\Program files\jdk*\bin\

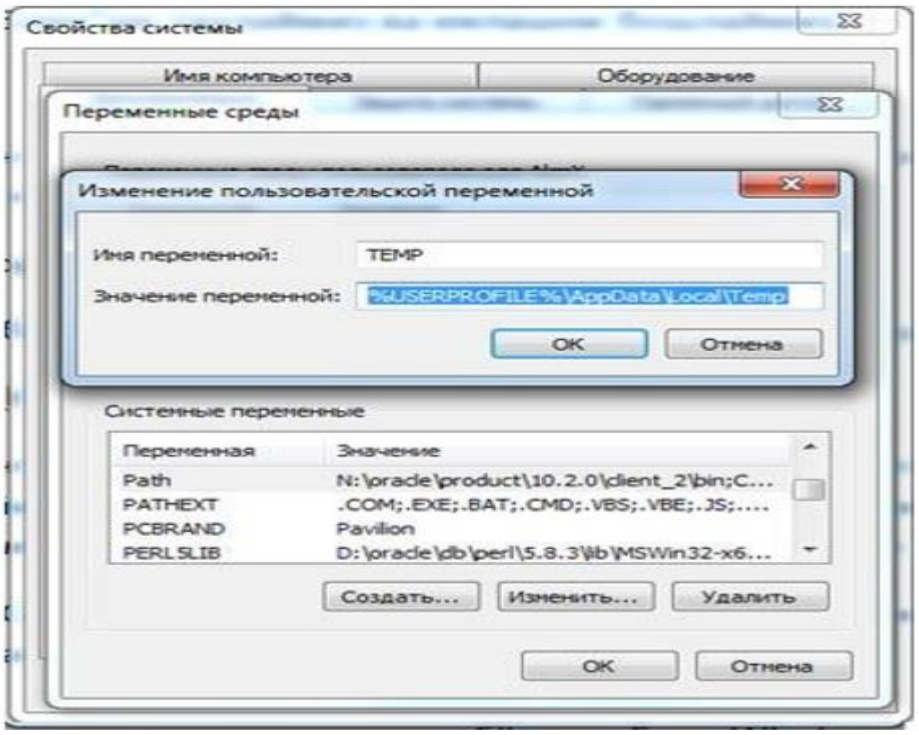
Bu adresda JDKning barcha bajariluvchi fayllari joylashgan.Ularni, operatsion tizimga ham ma'lum qilib qo‘yish lozim bo’ladi. Operatsion tizim bularni bilib olsa, bemalol "Командная строка" orqali ham ishlatish mumkin bo‘ladi.

Buni Windows 7 misolida ko‘rib chiqamiz.

Мой компьютер->Свойства, chap tomonda "Дополнительные параметры системы" bo‘limiga kiriladi va quyidagi oyna hosil bo’ladi.



Bu oynadan "Переменные среды" tugmasi bosiladi, hosil bo‘lgan oynaning "Системные переменные" bo‘limidan "PATH" o‘zgaruvchisini(переменная)qidirib, u belgilanadi va "Изменить" tugmasini bosiladi.



Keyingi bosqichda "Значение переменной"dagi qiymatlarning oxiriga o‘tiladi va "C:\Program Files\Java\jdk*\bin" adres kiritiladi(* o‘rniga o‘zingizning jdk adresingiz bo‘ladi, ya'ni mavjud papka nomi). "OK" tugmalari bosiladi, bu oynalardan chiqib ketiladi va kompyuter qayta yuklanadi.

Kompyuter qayta yuklanib bo’lgach endi shu kompyuterda javada dastur yozish imkoni mavjud boldi. JDKni o’rnatmasdan turib java kodlarini yozolmaymiz.

Kompyuterga JDK o’rnatilgandan so’ng Java kodlarini hatto termilal(CMD) dan ham ishga tushursa bo’ladi. Buning uchun avval biz java uchun biror bir kod yozamiz.

Misol uchun bloknot(notepad) ni ochamiz va quyidagi kodni yozamiz va uni asosiy klass (JavaDastur) ning nomi bilan bir xil nom bilan saqlaymiz.

```
class JavaDastur{
public static void main(String[] args){
 System.out.println("Salom Dunyo");
}
}
```

Yaratilgan java faylini kompiyatsiya qilish uchun termialni ochimiz va quyidagi amallarni bajaramiz.

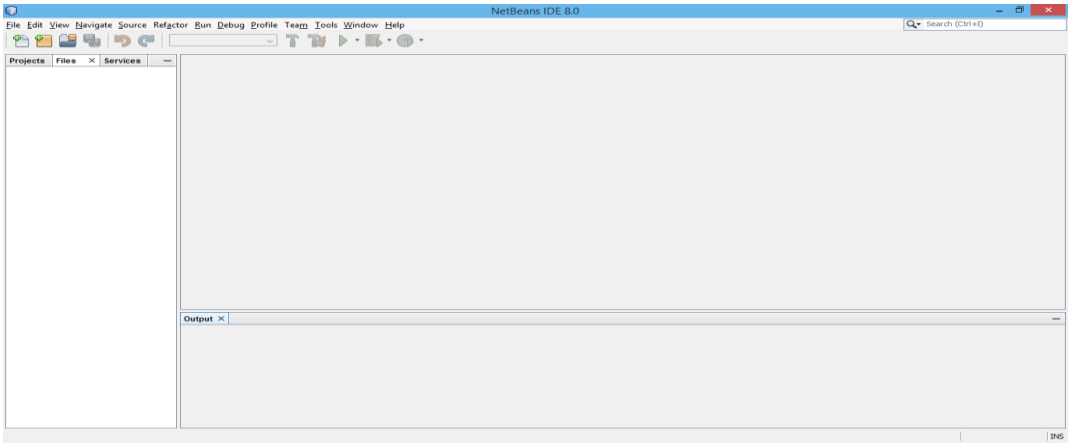
Natija yuqoridagidek ko’rinshda chiqariladi.

Javada tuzulgan eng oddiy dastur bo’lib buni terminal yordamida ishga tushini ko’rib o’tdik. Lekin shunday holatlar bo’ladiki murakkab dasturlarni tuzishga tog’ri keladi va ularni ishga tushirish uchun har safar terminaldan ishga turshurish biroz noqulay va dasturchini ko’p vaqtini oladi. Shuning uchun bunday holatlarning oldini olishga maxsus IDE lar ishlab chiqarilan bu IDE lar yordamida dastur ko’dlarini yozish ancha oson va ishga tushirishlikda ham qulay

hisoblanadi. Javada dastur tuzishda eng yaxshi IDE lar bular Eclipe va NetBeans hisobladadi.

NetBeansni yuklab olamiz va uni o’rnatamiz uni o’rnatish junda oddiy.

Netbensni o’rnatib bo’lganimizdan keyin uni ochamiz uning interfeysi quyidagicha.



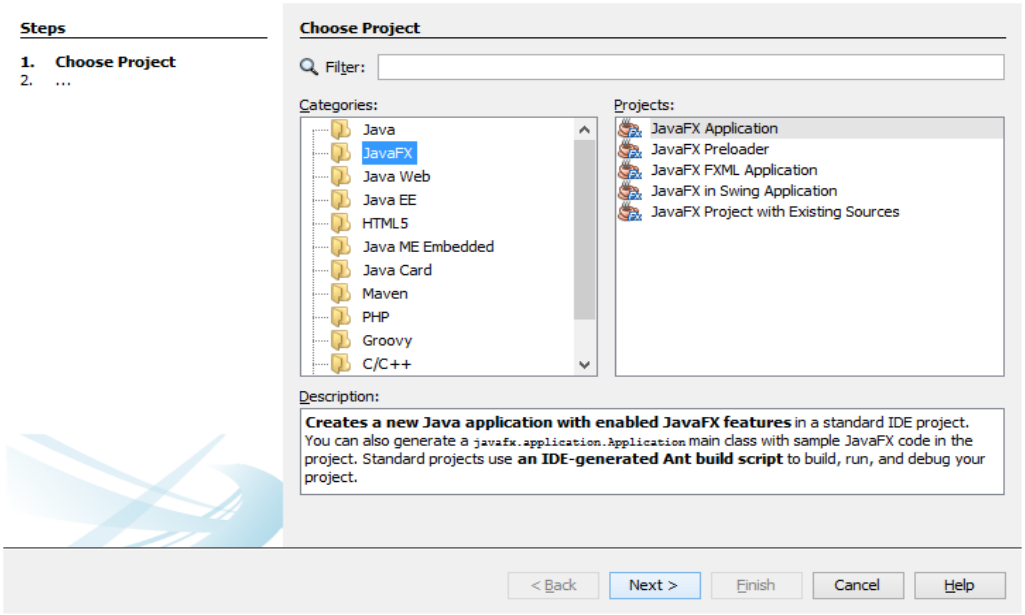
2.2.1-rasm.NetBeans IDE 8.0 ning interfeysi.

NetBeans IDE ning interfeysi juda sodda va qulay bo’lob u asosan 4 ta qismdan tashkil topgan.

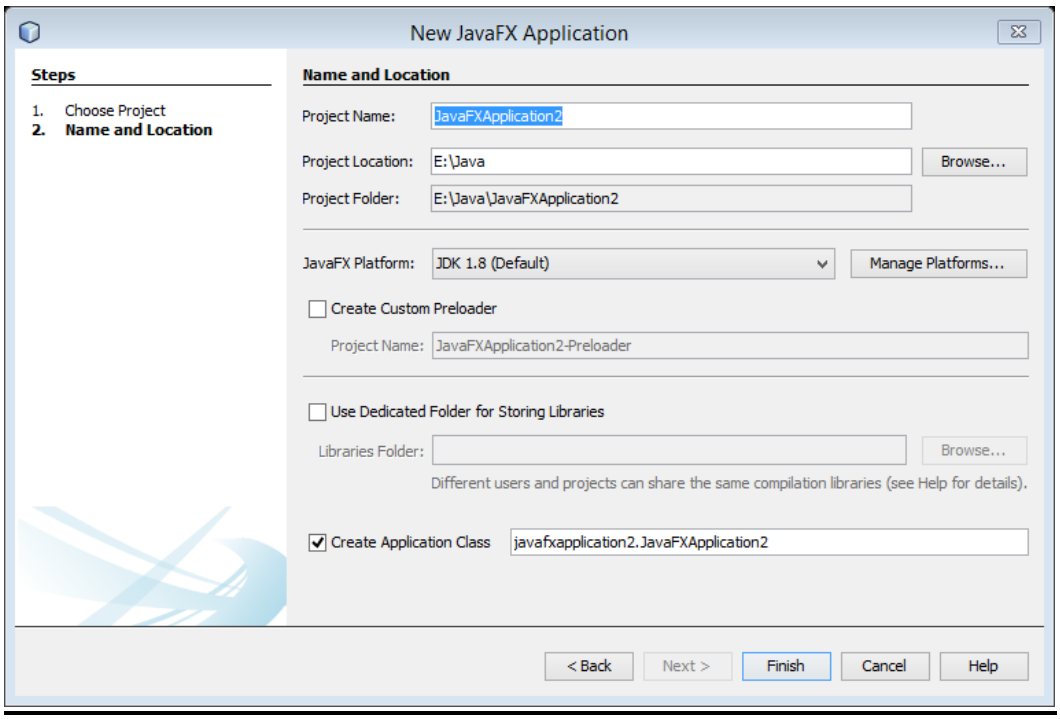
1. Boshqaruv paneli. Bu qismda hamma insturmentlar joylashgan.
2. Ochilgan fayllarni daraxt tartibida taxrilash paneli
3. Kod kiritich oynasi bu qism eng assoy bo’lib hisoblanadi bu yerga Javanink kodlari kiritladi.
4. Chiqarish paneli – bu qismda yozilgan kodlar natijasi chiqariladi ya’ni kompilyatsiya bo’lgan.

Netbens yodamida Java FX ni ishga tushurib ko’ramiz.

File -> New Project ni tanlaymiz va quyidagi oyna hosil boladi.



Categories bo'limidan **JavFX** ni tanlaymiz va Projects bolimidan **JavaFX Application**ni tanlaymiz va **Next** ni bosamiz va Keyingi oyna ochiladi.



Bu oynada proktini nomini kiritamiz, joylashuvini tanlaymiz va **Finish** tugmasini bosamiz va datur kodi yozish oynasi ochiladi.

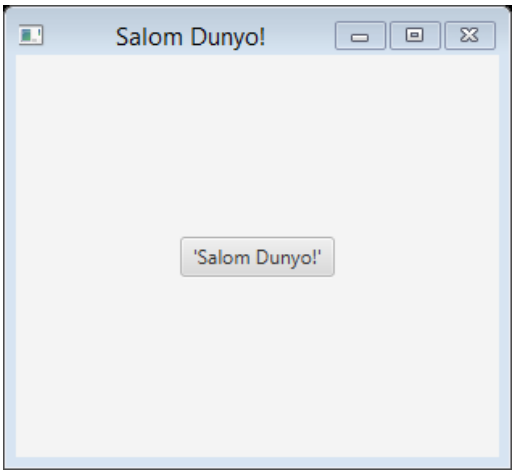
Java FX da oddiy Salom Dunyo so’zini chiqaruvchi dastur kodini yozib ko’ramiz. Dastur kodi quyidagicha.

```
package javafxapplication2;

import javafx.application.Application;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.layout.StackPane;
import javafx.stage.Stage;

/**
 *
 * @author Mahmud
 */
public class JavaFXApplication2 extends Application {

    @Override
    public void start(Stage primaryStage) {
        Button btn = new Button();
        btn.setText("Say 'Salom Dunyo!'");
        btn.setOnAction(new EventHandler<ActionEvent>() {

            @Override
            public void handle(ActionEvent event) {
                System.out.println("Salom Dunyo!");
            }
        });

        StackPane root = new StackPane();
        root.getChildren().add(btn);

        Scene scene = new Scene(root, 300, 250);

        primaryStage.setTitle("Salom Dunyo!");
        primaryStage.setScene(scene);
        primaryStage.show();
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        launch(args);
    }
```

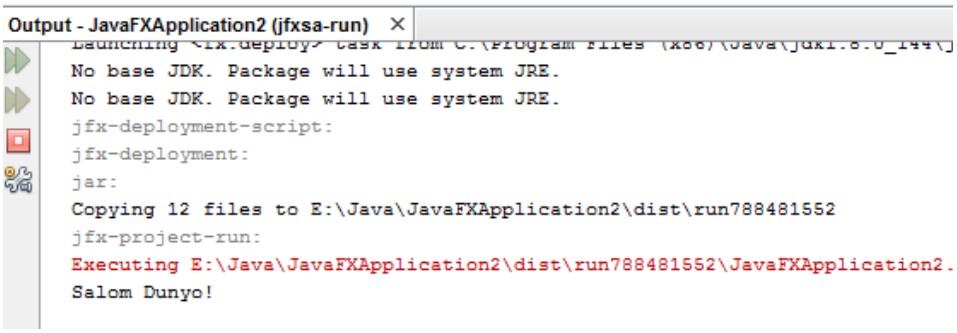
}

Natijasini ko’rish uchun **Run** buyrug’ini bosiladi yoki klaviaturadn **F6** tugmasi bosilsa ham bo’ladi.

Natija:



Ko’rinib turganidek Salom Dunyo ilovasi yaratildi va unda ‘Salom Dunyo’ tugmasi bor. Agar shu tugma bosilsa NetBean ichidagi chiqarish bo’limida quyidagicha chiqadi.



2.3. Matematik funksiyalar grafigini chizish dasturiy majmuasini ishlab chiqish.

Yangi proekt ochamiz va quyidagi kodlarni yozib boshlaymiz.

```
package magicalfunctiongenerator.aplicacio;

import magicalfunctiongenerator.domini.*;
import java.awt.*;
import java.text.*;
```

```
import java.util.*;
```

Kerakli kutuxonalrni qo’shib olgach ControladGrafica nomli klass yaratamiz.O’zgaruvchilarga qiymat beriladi. Bu klas orqali grafikani joylashadigan o’rni.

```
public class ControladorGrafica
{
  private Grafica grafica;
  private Mapper map;
  public static final DecimalFormat FORMAT_XIFRES = new
DecimalFormat("0.000");
  public static final int GRAFICA_CREADA = 0;
  public static final int FUNCIO_CREADA = 1;
  public static final int GRAFICA_ESBORRADA = 2;
  public static final int FUNCIO_ESBORRADA = 3;
  public static final int WORLD_COORDS_CHANGED = 4;
  public static final int SCREEN_COORDS_CHANGED = 5;
  public static final int FUNCIO_MODIFICADA = 6;
  public static final int GRAFICA_MODIFICADA = 7;
  public static final int ZOOM_MODIFICAT = 8;

  public static final int NO_COLISIO = 0;
  public static final int COLISIO_PUNT = 1;
  public static final int NO_COLISIO_PUNT = 2;
  public static final int COLISIO_FUNCIO = 3;
  public static final int NO_COLISIO_FUNCIO = 4;
  private boolean colisio;
  private Colisionable actual;
  private Funcio f;
  private float yvalue;
  public static final int FUNCIO_SIMPLE = 0;
  public static final int FUNCIO_INTERPOLADA = 1;
```

Konstruktor ochiladi va u string tipli o’zgaruvchilarni qabul qiladi.

```
  public ControladorGrafica(String grafica)
  {
    this.grafica                                          =
FabricaDAO.instancia().getRegistreGrafiques().get(grafica);
    map = this.grafica.getMapper();
  }

  public void dibuixar(Graphics2D g2)
  {
    grafica.dibuixar(g2);
  }
```

Funksiya yaratamiz setWorldCoords yomli unda o’zgaruvchiga beriladigan qiymatlarning maksimum va minimum qiymatlari qabul qiladi.

```
        public void setWorldCoords(float xmin, float xmax, float ymin,
float ymax)
        {
          map.setWorldCoords(xmin, xmax, ymin, ymax);
          grafica.canvis(WORLD_COORDS_CHANGED);
        }

        public void setScreenCoords(int xmin, int xmax, int ymin, int
ymax)
        {
          map.setScreenCoords(xmin, xmax, ymin, ymax);
          grafica.canvis(SCREEN_COORDS_CHANGED);
        }

        public float toWorldX(int x)
        {
          return map.toWorldX(x);
        }

        public float toWorldY(int y)
        {
          return map.toWorldY(y);
        }

        public float getXMinWorld()
        {
          return map.getXMinWorld();
        }

        public float getXMaxWorld()
        {
          return map.getXMaxWorld();
        }

        public float getYMinWorld()
        {
          return map.getYMinWorld();
        }

        public float getYMaxWorld()
        {
          return map.getYMaxWorld();
        }

        public void canvis(int canvi)
```

```
{
  grafica.canvis(canvi);
}

public float getDistY()
{
  return map.getDistY();
}

public float getDistX()
{
  return map.getDistX();
}

public int comprovarColisions(int x, int y)
{
  ListIterator colisionables = grafica.getColisionables();
  int tipus_colisio = NO_COLISIO;
  colisio = false;
  Colisionable actual2 = null;

  while (colisionables.hasNext())
  {
    actual2 = (Colisionable)colisionables.next();

    if (actual2.colisio(x, y))
    {
      colisio = true;
      actual2.setColisio(true);

      if (actual2 instanceof Funcio)
      {
        f = (Funcio)actual2;
        yvalue = f.getYValue(map.toWorldX(x));
        tipus_colisio = COLISIO_FUNCIO;
      }
      else if (actual2 instanceof Punt)
      {
        Punt p = (Punt)actual2;
        p.setImage(Punt.PIN2);
        tipus_colisio = COLISIO_PUNT;
      }

      break;
    }
    actual2.setColisio(false);
  }
```

```
          if (actual != null && (!actual2.getColisio() ||
(actual2.getColisio() && actual != actual2)))
          {
            actual.setColisio(false);
            if (actual instanceof Funcio)
            {
              tipus_colisio = NO_COLISIO_FUNCIO;
            }
            else if (actual instanceof Punt)
            {
              ((Punt)actual).setImage(Punt.PIN);   tipus_colisio   =
NO_COLISIO_PUNT;
            }
          }

          actual = colisio ? actual2 : null;

          return tipus_colisio;
        }

        public void incrPuntActual(float x, float y)
        {
          Punt p = (Punt)actual;
          p.setX(p.getX() + x);
          p.setY(p.getY() + y);
          FuncioInterpolada f = (FuncioInterpolada)p.getFuncio();
          f.firePuntsChanged();
          f.notificar(GestorDeCanvis.FUNCIO_MODIFICADA);
        }

        public int getTipusFuncioPuntActual()
        {
          Punt p = (Punt)actual;
          return p.getFuncio()instanceof FuncioSimple ? FUNCIO_SIMPLE
: FUNCIO_INTERPOLADA;
        }

        public void canviarColorFuncioActual()
        {
          ((Funcio)actual).setRandomColor();
        }


        public void esborrarPuntActual() throws Exception
        {
          Punt p = (Punt)actual;
          Funcio f = p.getFuncio();
          f.removePunt(p);
```

```
  if (f instanceof FuncioInterpolada)
  {
    ((FuncioInterpolada)f).firePuntsChanged();
  }
  grafica.canvis(GestorDeCanvis.GRAFICA_MODIFICADA);
}

public void afegirPuntFuncioActual(float x) throws Exception
{
  Funcio f = (Funcio)this.f;
  f.addPunt(x, f.getYValue(x));
  if (f instanceof FuncioInterpolada)
  {
    ((FuncioInterpolada)f).firePuntsChanged();
  }
  grafica.canvis(GestorDeCanvis.GRAFICA_MODIFICADA);
}

public int getYValueScreen()
{
  return map.toScreenY(yvalue);
}

public Color getColorColisio()
{
  if(actual instanceof Funcio)
    return ((Funcio)actual).getColor();
  else if(actual instanceof Punt)
    return ((Punt)actual).getFuncio().getColor();
  return Color.black;
}

public String getStringZoom()
{
  return FORMAT_XIFRES.format(getZoom()) + "%";
}

public float getZoom()
{
  return map.getDistX() * map.getDistY() / 400 * 100;
}

public String getStringColisio(int x, int y)
{
  if(actual instanceof Funcio)
    return formatCoordenades(toWorldX(x), toWorldY(y));
  else if(actual instanceof Punt)
  {
    Punt p = (Punt) actual;
```

```
            return formatCoordenades(p.getX(), p.getY());
          }
          return "";
        }

        public void eliminarFuncioActual()
        {
          try
          {

GestorDeCanvis.instancia().removeFuncio(((Funcio)actual).getExpressi
o(), grafica.getNom());
            GestorDeCanvis.instancia().notificar(grafica.getNom(),
GestorDeCanvis.FUNCIO_ESBORRADA);
            actual = null;
          }
          catch(Exception e){}
        }


        public static String formatCoordenades(float x, float y)
        {
          return   "("   +   FORMAT_XIFRES.format(x)   +   "   "   +
FORMAT_XIFRES.format(y) + ")";
        }

        static
        {
          DecimalFormatSymbols symbols = new DecimalFormatSymbols();
          symbols.setDecimalSeparator('.');
          FORMAT_XIFRES.setDecimalFormatSymbols(symbols);
        }
      }
```

Yangi oyna ochib asosiy formamizni yaratib olami.

```
package magicalfunctiongenerator.aplicacio;

import magicalfunctiongenerator.domini.*;
import java.awt.*;
import java.util.*;
```

GestorDeCanvis nomli asosiy klass yaratib olamiz va o’zgaruvchilarni public va privite tarzda elon qilamiz.

```
public class GestorDeCanvis
{
  private int numGrafiquesCreades = 0;
  private static GestorDeCanvis instancia;
  private IRegistreGrafiques grafiques;

  public static final int GRAFICA_CREADA = 0;
  public static final int FUNCIO_CREADA = 1;
  public static final int GRAFICA_ESBORRADA = 2;
  public static final int FUNCIO_ESBORRADA = 3;
  public static final int WORLD_COORDS_CHANGED = 4;
  public static final int SCREEN_COORDS_CHANGED = 5;
  public static final int FUNCIO_MODIFICADA = 6;
  public static final int GRAFICA_MODIFICADA = 7;
  public static final int ZOOM_MODIFICAT = 8;
```

Konstuktor yaratamiz va uning ichiga quyidagi shartni kirtamiz.

```
 public static GestorDeCanvis instancia()
  {
    if (instancia == null)
    {
      instancia = new GestorDeCanvis();
    }
    return instancia;
  }

  private GestorDeCanvis()
  {
    grafiques = FabricaDAO.instancia().getRegistreGrafiques();
  }

  public void addObservador(Observer o, String grafica)
  {
    Grafica g = grafiques.get(grafica);
    g.addObserver(o);
  }

  public void addFuncio(String funcio, String grafica) throws
Exception
  {
    Grafica g = grafiques.get(grafica);
    g.addFuncio(funcio);
    g.canvis(FUNCIO_CREADA);
  }

  public void addFuncio(float[] x, float[] y, String grafica) throws
Exception
```

```
  {
    Grafica g = grafiques.get(grafica);
    g.addFuncio(x, y);
    g.canvis(FUNCIO_CREADA);
  }

  public void addGrafica(String nom, LinkedList observers, float
xminw, float xmaxw, float yminw, float ymaxw, int xpmins, int xpmaxs,
int ypmins, int ypmaxs) throws Exception
  {
    Grafica g = new Grafica(nom, xminw, xmaxw, yminw, ymaxw, xpmins,
xpmaxs, ypmins, ypmaxs);
    grafiques.add(g);

    ListIterator list = observers.listIterator();
    while (list.hasNext())
    {
      g.addObserver((Observer)list.next());

    }
    numGrafiquesCreades++;
    g.canvis(GRAFICA_CREADA);
  }

  public void setNotificar(String grafica, boolean notificar)
  {
    getGrafica(grafica).setNotificar(notificar);
  }

  public void setCoordenadesMon(String grafica, float xmin, float
xmax, float ymin, float ymax)
  {
    Grafica g = getGrafica(grafica);
    g.setWorldCoords(xmin, xmax, ymin, ymax);
    g.canvis(WORLD_COORDS_CHANGED);
  }

  public void setCoordenadesPantalla(String grafica, int xmin, int
xmax, int ymin, int ymax)
  {
    Grafica g = getGrafica(grafica);
    g.setScreenCoords(xmin, xmax, ymin, ymax);
    g.canvis(SCREEN_COORDS_CHANGED);

  }

  public void setColor(String funcio, String grafica, Color c)
  {
    grafiques.get(grafica).getFuncio(funcio).setColor(c);
```

```
  }

public void removeFuncio(String funcio, String grafica) throws
Exception
  {
    Grafica g = getGrafica(grafica);
    g.removeFuncio(funcio);
    g.canvis(FUNCIO_ESBORRADA);
  }

  public void removeGrafica(String grafica) throws Exception
  {
    Grafica g = getGrafica(grafica);
    if (g == null)
    {
      throw new Exception("The graphic cannot be deleted because it
doesn't exist.");
    }
    grafiques.remove(grafica);
    g.canvis(GRAFICA_ESBORRADA);
  }

  public void removeObservador(Observer o, String grafica)
  {
    Grafica g = grafiques.get(grafica);
    g.deleteObserver(o);
  }

  private Grafica getGrafica(String grafica)
  {
    return grafiques.get(grafica);
  }

  public float getDistanciaMaxima()
  {
    return Mapper.DISTANCIA_MAXIMA;
  }

  public float getXifraMaxima()
  {
    return Mapper.XIFRA_MAXIMA;
  }

  public float getXifraMinima()
  {
    return Mapper.XIFRA_MINIMA;
  }

  public int getNumeroGrafiquesCreades()
```

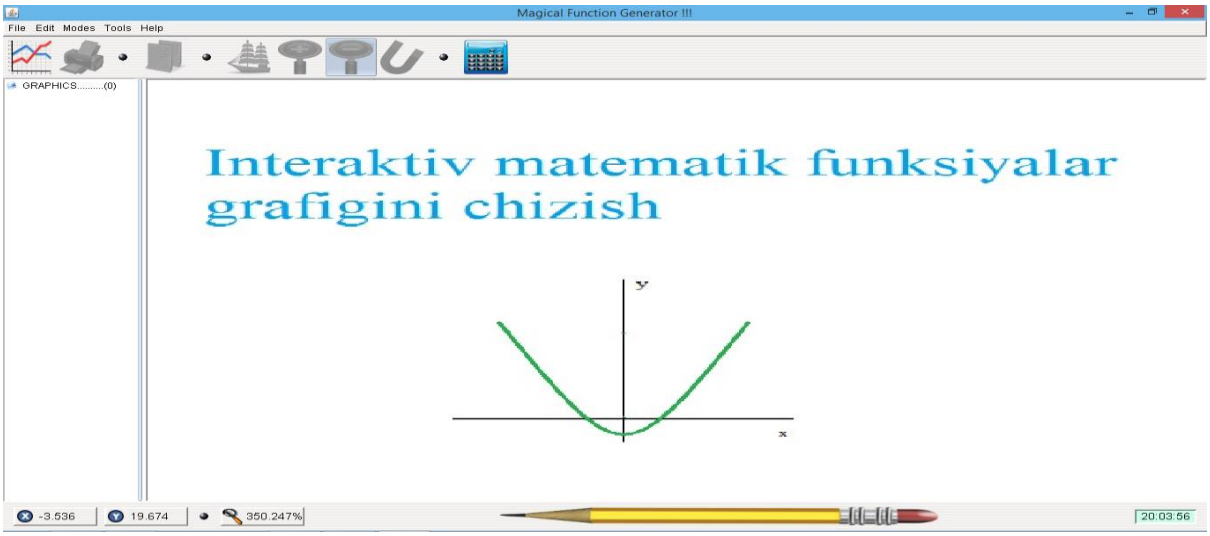
```
{
  return numGrafiquesCreades;
}

public int getXMaxScreen(String grafica)
{
  return grafiques.get(grafica).getMapper().getXMaxScreen();
}

public int getYMaxScreen(String grafica)
{
  return grafiques.get(grafica).getMapper().getYMaxScreen();
}

public ListIterator getGrafiques()
{
  return grafiques.getGrafiques();
}

public ListIterator getFuncions(String grafica)
{
  return grafiques.get(grafica).getFuncions();
}

public String getGraficaIndex(int index)
{
  return grafiques.getGraficaIndex(index);
}

public int getNumeroFuncionsGrafica(String grafica)
{
  return getGrafica(grafica).getNumeroFuncions();
}

public int getNumeroGrafiques()
{
  return grafiques.size();
}

public String getFuncioIndex(int index, String grafica)
{
  return getGrafica(grafica).getFuncioIndex(index);
}

public int getIndexGrafica(String grafica)
{
  return grafiques.getIndex(grafica);
}
```

```
  public int getIndexFuncioGrafica(String funcio, String grafica)
  {
    return getGrafica(grafica).getIndexFuncio(funcio);
  }


  public boolean existeix(String grafica)
  {
    return getGrafica(grafica) != null;
  }

  public void notificar(String grafica, int canvi)
  {
    getGrafica(grafica).canvis(canvi);
  }
}
```

2.4. Interaktiv matematik funksiyalar grafigini chizish dasturiy majmuasi bilan ishlash

Matematik funksiyalar bilan ishlaganda uning grafigini chizib ko'rishga zarurat paydo bo'lib qoladi shunday paytda matematik funksiyalar grafigini chizuvchi dasturiy vositadan foydalanish qulaylik yaratadi bunda funksiyani grafigi to'liq aks etiradi va matematik funksiyalar bilan ishlaganda uni to'liq tushunish imkonini yaratadi.

Dastur interfeysi quyidagicha:



2.4.1-rasm.Dastur interfeysi

Dastur interfeysi asosan 4 ta qismdan tashkil topgan.

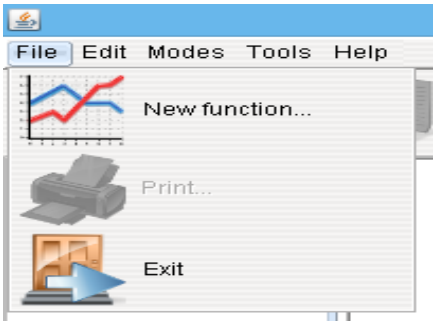
1.Toolbar yani dasturning yuqori qismi.
2.Graphics list yani chizilgan grafiklar ro'yxatlarini aks ettirb turuvchi oyna

3.Asosiy oyna bu qismida chiqizlgan grafiklar oynalari chiqadi.
4.Pastki qism.

Dasturning yuqri qismida 5 ta bo’lim va komponentalar joylashgan.

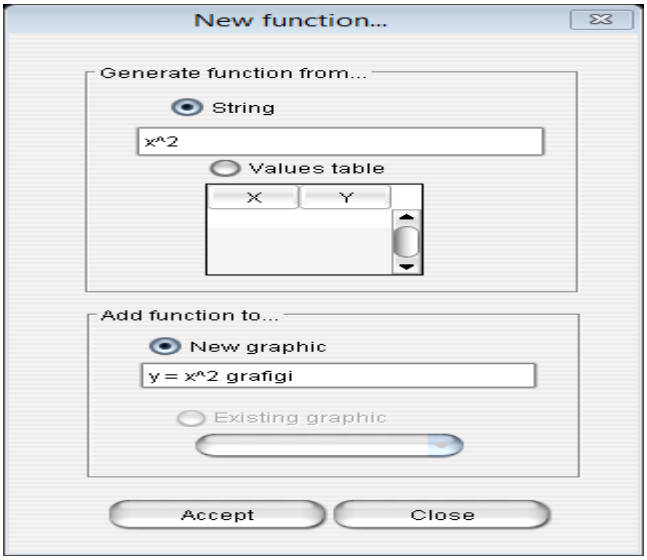
2.4.2-rasm.Dasturnig yuqori qismi.

1.File – bu bo’limda 3 hususiyat joylashgan bo’lib ular new function, print, exit, tugmalaridir.



2.4.3-rasm.Dasturning file menyusi

New function – bu tugmani bosib yangi funksiya chizish uchun qiymat kiritish oynasi ochiladi.

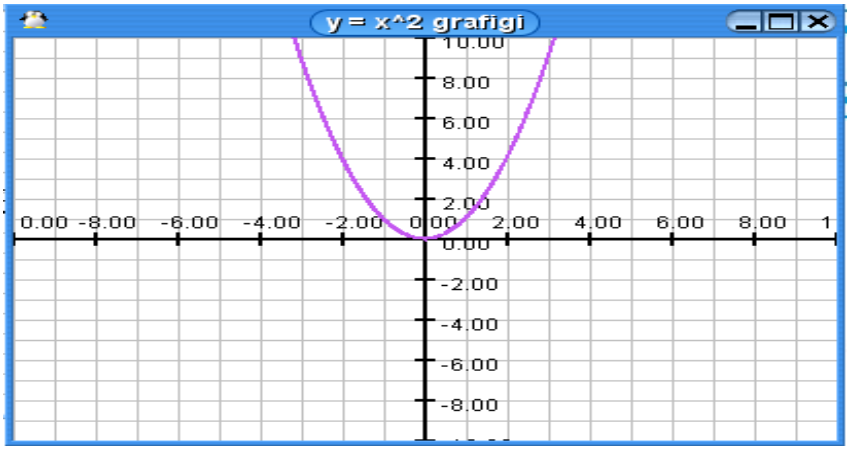


2.4.4-rasm.Funksiya kiritish formasi.

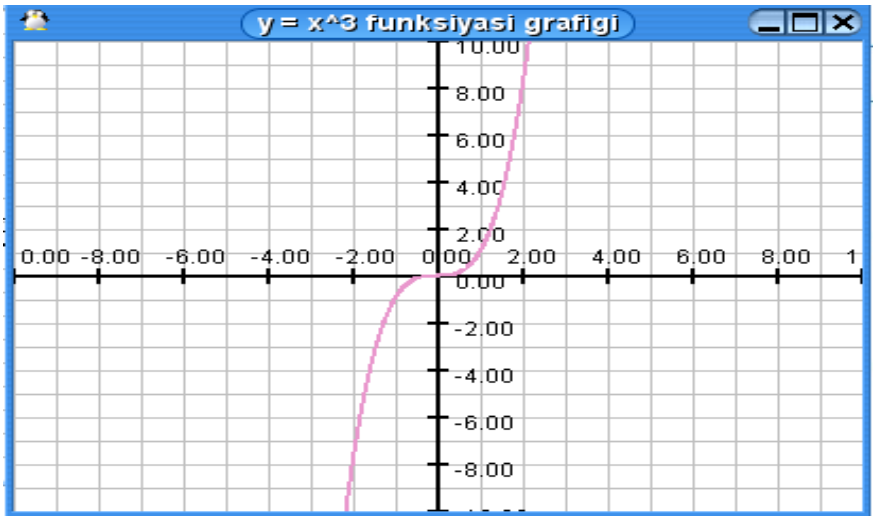
Bu formada funksiyani 2 xil usulda kiritish mumkin. 1-usulda to’g’ri funksiyaning o’zi kiritiladi va funksiyaga nom beriladi va **Accept** tugmasi bosiladi. 2-usulda esa hohlagan turdagi funksiyani qiymatlarini berish orqali

chizish mumkin. Bu usuldan foydalanganda yuqorida joylashgan kalkulyotordan foydalansa ancha oson bo’ladi.

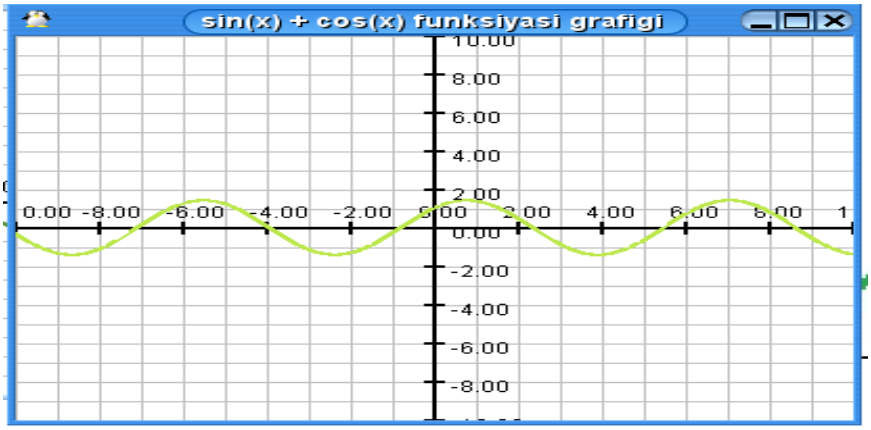
1-usulda chizilgan funksiyaning grafigi quyidagicha bo’ladi.



2.4.5-rasm.Y = x^2 funksiyasi grafigi.

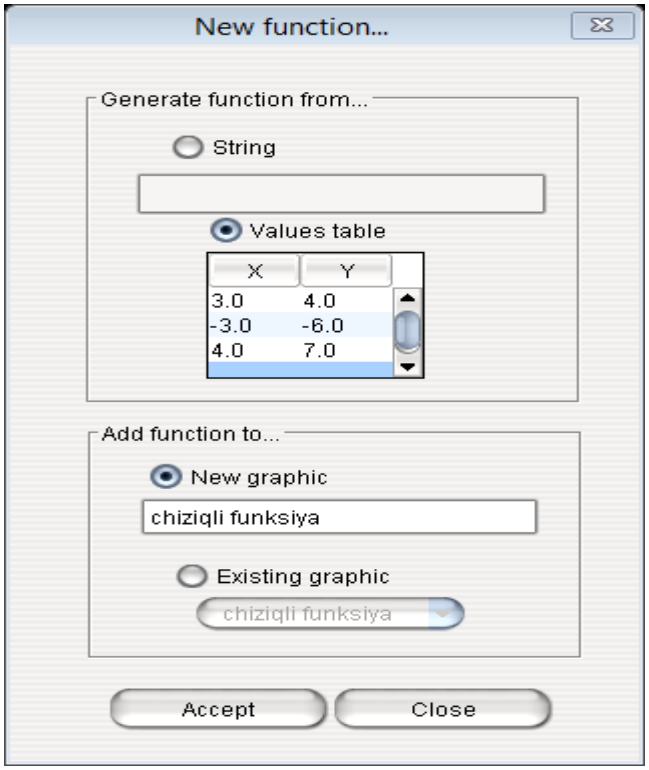


2.4.6-rasm. Y = x^3 funksiyasi grafigi.



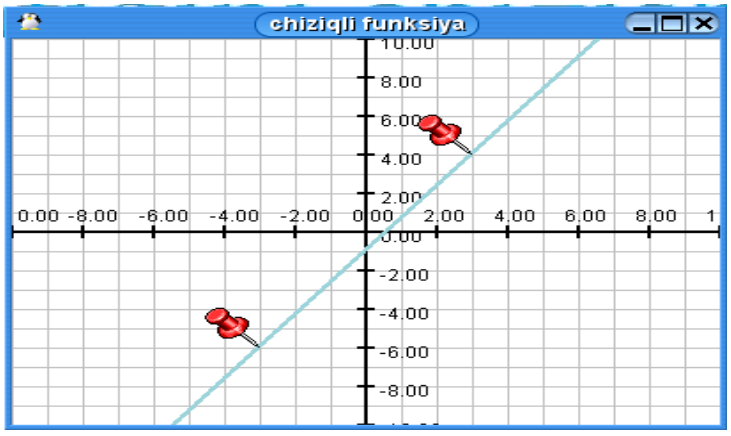
2.4.7-rasm.cos(x) + sin(x) funksiyasi grafigi

2-usulda grafik chizishni ko’rib chiqamiz.Buning uchun **Value table** tugmasini tanlab qo’yamiz va funksiya kiritish oynasidan funksyaning nomalum qiymatlariga son berib chiqamiz.



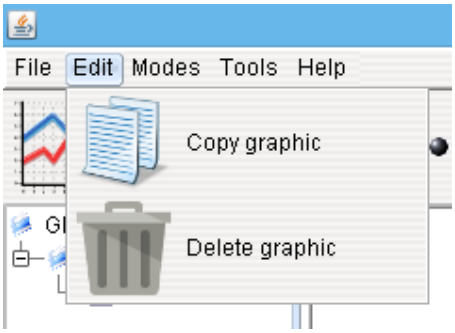
2.4.8-rasm.Funksiya grafigini qiymat berish orqali chizish.

Natija quyidagicha:



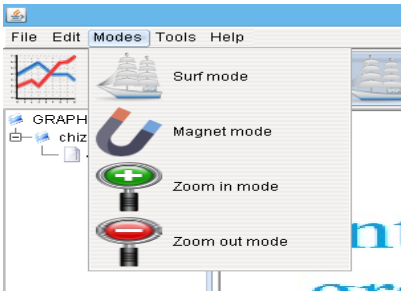
2.4.9-rasm.Qiymat berish orqlali chizilgan funksiya grafigi

2.Edit – bu bo’lim 2 ta qismdan tashkil topgan bolib, unda funksiya grafigini nusxalash va o’chirib tashlash mukin.



2.4.10-rasm.Edit bolimi.

3.Modes – bu bo'lim 4 ta qismdan tashkil topgan bo'lib **surf mode** tugmasi orqali kursor ko'rinishi o'zgartirish mumkin. **Magnet mode** tugmasi orqali esa grafikdagi berildan qiymatlarning sonii 3 xona aniqlikda ko'rish mumkin, **Zoom in mode** va **Zoom out mode** tugmalari orqali grafikani kattalashtirish va kichilashtirish mumkin.



2.4.11-rasm.Edit bolimi.

4.Tools - bu bo'limda faqat kalkulyator joylashdan u qiymat berishda ishlatiladi.

5.Help - bu bo'lim ichida **about** tugmasi bo'lib u yordamida dastur haqida ma'lumot olish mumkin.

XULOSA

Hozirgi vaqtda axborot texnologiyalarini ta`lim sohalarida qo'llash bugungi kunda juda muhim bo'lib sanaladi.Axborot texnologiyalarini ta`lim sohalarida qo'llash orqali murakkab masalarni oddiy va o'quvchilar tushunadigan tarzda yechish imkoni osonlashadi. Ushbu birituv malakaviy ishda asosan biror bir matematik funksiyani yechishda uni grafigini tasvirlab berish orqali o'quvchi bu masalaning to'liq moxiyatini anglab yetishi maqsad qilingan. Matematik funksiyalar grafigini chizishni o'rganib chiqish va uni amalda dasturlash tillari yordamida qo'llanish maqsad qilib qo'yilgandi va bu maqsadga erishildi.

Mazkur bitiruv malakaviy ishi kirish, 2 ta bob va xulosadan iborat. Kirish qismida mavzuning dolzarbligi va maqsadi ochib berildi.

Birinchi bobda matematik funksiyalar grafigini chizish dasturiy vositalar tahlil qilib chiqildi.Buning uzun kitoblardan va Internet manbalaridan foydlanildi.

Ikkinchi bobda Java dasturlash tilida interaktiv matematik funksiyalar grafigini chizish va dasturiy majmuasini ishlab chiqildi.

Qo'yilgan maqsadga erishish uchun quyidagi vazifalar bajarildi:

- Matematik funksiyalar grafigini chizish tahlil qilib chiqildi;
- Matematik funksiyalar chizadigan dasturiy muhitlar bilan tanishib chiqildi;
- Tekislikda grafik chizish usullarini o'rganib chiqildi;
- Diagramma va Gistagrammalarni chizishni o'rganib chiqildi;
- Java dasturlash tili bilan tanishib chiqildi;
- Java dasturlash tiling grafik imkoniyataridan foydalanib dastur tuzildi;
- Interaktiv funksiyalar grafigini chizishni Java tilida ishlab chiqildi.

Bitiruv malakaviy ishini bajarish davomida qo'yilgan maqsadlarga to'liq erishildi va ijobiy natijalar olindi.

FOYDALANILGAN ADABIYOTLAR

1.Funksiya va grafikalar. A.Gaziyev, I.Isroilov, M, Yaxshiboyev.

2. utilizing microsoft mathematics in teaching and learning calculus

Rina Oktaviyanthi, Yani Supriani Universitas Serang Raya, Jl. Raya Serang-Cilegon Km 5, Taman Drangong, Serang, Banten 42162 e-mail: rinaokta1210@yahoo.com

3. Duval. (2012). Using Computer Technology in Teaching and Learning Mathematics in an Albanian Upper Secondary School. Kristiansand: University of Agder

4. Дьяконов В. П. Maple 7. Учебный курс. СПб.:ПИТЕР, 2002. 41-43б

5. Дьяконов В. П. Maple 8 в математике, физике и образовании. М.: СОЛОН-Пресс, 2003. 44-48б

6. P. Nouton, G.Shildt Java

INTERNET MANBALARI

1.https://en.wikipedia.org/wiki/Microsoft_Mathematics

2.https://Tami.uz/sayt/tami.uz/matnga_qarang1d56.html?id=323

3.www.java2s.com

4.https://ru.wikipedia.org/wiki/JavaFX

5.https://www.mathworks.com/