O‘ZBEKISTON RESPUBLIKASI AXBOROT TEXNOLOGIYALARI VA KOMMUNIKATSIYALARINI RIVOJLANTIRISH VAZIRLIGI

MUHAMMAD AL-XORAZMIY NOMIDAGI TOSHKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI

“ALGORITMLASH VA MATEMATIK MODELLASHTIRISH” KAFEDRASI

“ALGORITMLARNI LOYIHALASH”

Fanidan laboratoriya topshiriqlarini bajarish bo‘yicha uslubiy ko‘rsatmalar

TOSHKENT 2021

Muhammad Al-Xorazmiy nomidagi Toshkent axborot texnologiyalari universiteti “Dasturiy injiniring” fakulteti “Algoritmlash va matematik modellashtirish” kafedrasi 2021 y.

**Mualliflar: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ .**

“Algoritmlarni loyihalash” fanidan laboratoriya topshiriqlarini bajarish bo‘yicha uslubiy ko‘rsatmalar”, -Toshkent: MuhammadAl-Xorazmiy nomidagiTATU. 2021. -\_\_\_b.

Ushbu o‘quv-uslubiy ko‘rsatma TATU da o‘qitilayotgan “Algoritmlarni loyihalash” fanidan laboratoriya ishlarini bajarishda foydalanish uchun mo‘ljallangan bo‘lib, talabalarda bu fan haqidagi bilim va ko‘nikmalarni rivojlantirishga xizmat qiladi.

Muhammad Al-Xorazmiy nomidagi Toshkent axborot texnologiyalari universiteti ilmiy-uslubiy Kengashining qarori bo‘yicha chop etilgan. (\_\_.\_\_.2021yil, \_\_ -sonli bayonnoma)

Muhammad Al-Xorazmiy nomidagi Toshkent axborot texnologiyalari universiteti.

**SO‘Z BOSHI**

Fan va texnika jadal rivojlanayotgan hozirgi davrda, bu taraqqiyot tufayli vujudga kelayotgan ilmiy va texnik muammolar ham keskin ortib bormoqda. Ularni yechish uchun esa bu masalalar matematik modeliga mos keladigan aniq yoki taqribiy usullarni yaratish yoki tanlash zarurati paydo bo‘ladi. Bu usullar bo‘yicha hisoblashlar esa katta hajmli bo‘lib, ularni bajarishda kompyuterlardan foydalanishga to‘g‘ri keladi. Buning uchun esa mazkur usullar algoritmi bo‘yicha tuzilgan dasturlar kerak bo‘ladi. “Algoritmlarni loyihalash” fani bo‘yicha o‘tilgan ma’ruza va amaliy mashg‘ulotlar davomida bir qator ma’lum va keng qo‘llaniladigan sonli usullar haqida ma’lumotlar berilib, oddiy misollarda bevosita hisoblash yo‘li bilan talqin qilinadi. Bu usullarni keng qamrovli, jiddiy hollarda tadbiq qilish saviyasi va ko‘nikmasini shakllantirish uchun talabalarga har biri uchun shaxsiy topshiriqlar beriladi. Bu topshiriqlarni bajarish, tahlil qilish va himoya qilish jarayonida talabada, avvalo, kompyuterdan foydalanish malakasi oshadi, undan foydalanish samaradorligiga ishonch paydo bo‘ladi. Eng asosiysi, talabada o‘ziga ishonch va ijodiy fikrlash qobiliyati shakllanadi. Mazkur uslubiy ko‘rsatma mavjud va tavsiya qilingan adabiyotlarga qo‘shimcha bo‘lib, bevosita laboratoriya mashg‘ulotlariga mo‘ljallangan. Qo‘llanma fan bo‘yicha o‘quv dasturiga to‘la mos keladi. Unda har bir mavzu bo‘yicha qisqacha nazariy va amaliy ma’lumotlar berilib, shu mavzu bo‘yicha talabalar bajarishi kerak bo‘lgan shaxsiy topshiriq variantlari berilgan. Qulay bo‘lishi uchun har bir usul bo‘yicha namunaviy topshiriqni bajarish tartibi va dasturi bo‘yicha ham yetarli ma’lumotlar berilgan. O‘ylaymizki, ushbu o‘quv uslubiy ko‘rsatma, nafaqat talabalar, balki bu yo‘nalishga qiziqqan barcha mutaxassislar uchun ham foydali bo‘ladi.

**Laboratoriya ishi №1.**

**CHiziqli, tarmoqlanuvchi va takrorlanuvchi algoritmlar.**

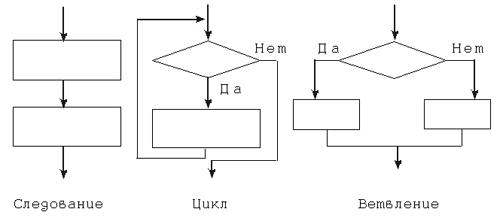
***Ishdan maqsad: Talabalarda algoritmlarni asimptotik tahlil qilish haqida ko’nikmalar hosil qilish, masalalarni yechishda saralash, qidirish algoritmlarini qo’llash va ularni tahlil qilish orqali qulayini tanlash.***

***Nazariy qism:***

**Algoritm** – berilgan natijaga erishish uchun qilinishi kerak boʻlgan aniq koʻrsatmalar ketma-ketligi. **Algoritm** keng maʼnoda faqat kompyuterga oid atama boʻlmay, balki unda berilgan koʻrsatmalarni bajara oluvchi har qanday narsaga oiddir.

**Algoritm**, algorifm – maʼlum bir turga oid masalalarni yechishda ishlatiladigan amallarning muayyan tartibda bajarilishi haqidagi aniq qoida (dastur). Kibernetika va mat.ning asosiy tushunchalaridan biri. O‘rta asrlarda sanoqning o‘nli tizimi bo‘yicha to‘rt arifmetik amal bajariladigan qoidani A. deb atashgan. "Bu qoidalarni mat.ga 9-asrda al-Xorazmiy kiritgan. Yevro-pada bunday qoidalar uning tugilgan yurtiga nisbatan lotinchalashtirilgan (Algoritmus yoki Algorithmus shaklida "algorizm" deyilgan), keyinchalik "algoritm"ga aylangan" (akad. A. N. Kol-mogorov). Fanda "Yevklid algoritmi", "G‘iyosiddin Koshiy algoritmi", "Laure algoritmi", "Markov algoritmi" deb ataluvchi A.lar maʼlum. A. tushunchasi tobora kengayib borib, kibernetikaning nazariy va mantiqiy asosi hisoblangan A.lar nazariyasi paydo bo‘ldi. Oʻzbekiston Respublikasi da bir necha ilmiy tadqiqot muassasalari va hisoblash mar-kazlarida A.dan foydalanish sohasida samarali ishlar olib borilmoqda. Mas, O‘zbekiston Fanlar Akademiyasi "Kibernetika" ilmiy ishlab chiqarish birlashmasida, O‘zbekistondagi bar-cha universitetlarda, Toshkent davlat texnika untida, Oʻzbekiston Respublikasi Makroiqgisod va statistika vazirligi qoshidagi Hisoblash markazi va boshqa muassasalarda olib borilayotgan ishlar bunga misol bo‘la oladi.

**Algoritmning ushta turi mavjud: shiziqli, tarmoqlanuvshi va takrorlanuvshi.**



Chiziqli algoritmlar - hesh qanday shartsiz faqat ketma-ket bajariladigan jarayonlardir.

Tarmoqlanuvshi algoritmlar - ma’lum shartlarga muvofiq bajariladigan jarayonlardir.

Takrorlanuvshi algoritmlar - biron bir shart tekshirilishi yoki biron parametrning har xil qiymatlari asasidachekli ravishda takrorlanish yuz beradigan jarayonlardir.

***Chiziqli algoritmlarni blok-sxemasi***

Hech qanday shart tekshirilmaydigan va tartib bilan faqat ketma – ket bajariladigan algoritmlar

***ch i z i q l i a l g o r i t m l a r*** deb yuritiladi .

kiritish

Chop etish

Qiymatlarni o’zlashtirish

Chiziqli algoritmlar va dasturlar odatda juda sodda masalalarni yechiщda qo‘llaniladi. Bu masalalar yechimi biror shartga yoki siklik amallar bajarilishiga bog‘liq emas.

Masalan, to‘g‘ri to‘rtburchakning tomonlariga ko‘ra uning perimetri, diagonali va yuzasini hisoblashni (a, b – tomonlar qiymatiga ko‘ra) quyidagicha tashkillashtirish mumkin.

a,b-lar qiymatini kiritish

p:=2\*a+2\*b

d:=sqrt(sqr(a)+sqr(b))

s:=a\*b

p, d, s

Yechish:

*//Muallif: Begimov Uktam*

*//Sana: 25.01.2021 yil*

*//Maqsad:* To‘rtburchak yuzi hisoblash

*#inclide <iostream>*

*using namespace std;*

*int main()*

*{*

*float a, b;*

*cout <<”A tomonning qiymati kiritilsin=”; cin >>a;*

*cout <<”B tomonning qiymati kiritilsin=”; cin >>b;*

*P=2\*a+2\*b;*

*D=sqrt(sqr(a)+sqr(b));*

*S=a\*b;*

*cout <<”to‘rtburchak perimetri=”<<P<<endl;*

*cout <<”to‘rtburchak ioganperli=”<<D<<endl;*

*cout <<”to‘rtburchak yuzasi =”<<S<<endl;*

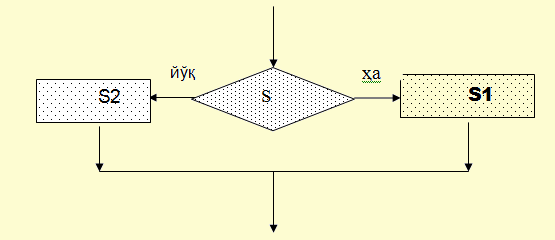
*return 0;*

*}*

**Tarmoqlanish va o‘tish operatorlari.**

Turli masalalarni yechganda ko‘rsatmalarni bajarish tartibi biror bir shartning bajarilishiga bog‘liq holda bajariladi. Ya’ni algoritm tarmoqlanadi. Tarmoqlanish «echim» bloki orqali ifodalanadi.

Ma’lum bir shartni bajarilishi yoki bajarilmasligiga qarab, tarmoqlanuvchi jarayon holatlari aniqlanadi. Tarmoqlanuvchi jarayonlarni hisoblash uchun shartli operatordan foydalaniladi. Shartli operator ikki xil ko‘rinishda bo‘ladi:



Shartli operatordan foydalanishga misollar keltiramiz.

1-misol. Kiritilgan ixtiyoriy butun sonni juft yoki toqligini aniqlovchi dastur yarating.

// Mu’allif: Begimov Uktam

// butun sonni juft yoki toqligini aniqlovchi dastur

#include <iostream>

using namespace std;

int main ()

{

int a;

cin >> a;

if (a % 2 ==0)// ‘if x mod 2 =0’ boshqa dasturlash tillarida

{

cout <<”juft”;

}

else

{

cout <<”toq”;

}

return 0;

}

**Takrorlash operatori.**

Yechilayotgan masalaning mohiyatiga qarab, dasturchi tuzuvchi o‘zi uchun qulay bo‘lgan takrorlash operatorini tanlab olishi mumkin.

Takrorlash operatorlarining 3 xil turi mavjud:

·      parametrli takrorlash operatori;

·      repeat  takrorlash operatori;

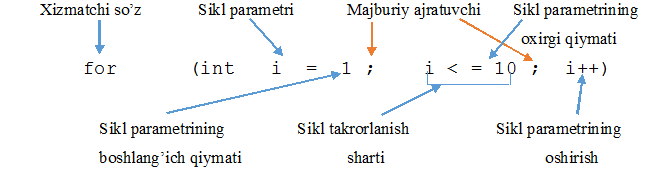
·      while takrorlash operatori.

1-misol:

For i:=1 to 23 do

s:=s+1/I;

Siklning bu holatida parametr i-ning qiymati dastlab 1-ga teng bo‘lib, sungra siklning har bir qadamida ‘+1’-ga orta boradi va 2,3,…,23 ga teng bo‘ladi. Zarur hollarda parametrning qiymatini ‘-1’ ortttirish mumkin bo‘lib, bunda «to» o‘rniga «downto» ishlatiladi.



**Misol.** *Sonli massiv  A = (a1 , a2 , ... , aN ) ning elementlarini yig’indisini hisoblang.*

**Test**

|  |  |  |
| --- | --- | --- |
| Berilgan | | Natija |
| N=5 | A=(3, 5, -2, 6, 3) | S=15.0 |

Начало формы

Конец формы

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Algoritmi:*  #include <iostream>  #include <math.m>  using namespace std;  int main ()  {  Int a[3, 5, -2, 6, 3];  for (float i = 1; i <=50; i++)  s=i++;  cout <<s<<endl;  return 0;  } | | *Algoritmning bajarilishi*   |  |  | | --- | --- | | i | S | |  | 0 | | 1 | *0 + a1 = 0+3 = 5* | | 2 | *a1 + a2 = 3+5 = 8* | | 3 | *a1+a2+a3 = 8-2 = 6* | | 4 | *a1+a2+a3+a4 = 6+6 = 12* | | 5 | *a1+a2+a3+a4+a5 = 12+3=15* | | |
| **alg** Summa (**but** N,  **haqjad** A[1:N], **haq** S)  **arg** N,A  **boshlbut** i    S:=0  **sb** i **uchun** 1 **dan** N **gacha**    S := S + A[i]  **so**  **tamom** | | *Blok sxemasi:* | |

Quyidagi yig’indini hisoblovchi dastur tuzing.

//’Takrorlanuvchi operatori’

#include <iostream>

#include <math.m>

using namespace std;

int main ()

{

float S=0;

for (float i = 1; i <=50; i++)

s+=1/i;

cout <<s<<endl;

return 0;

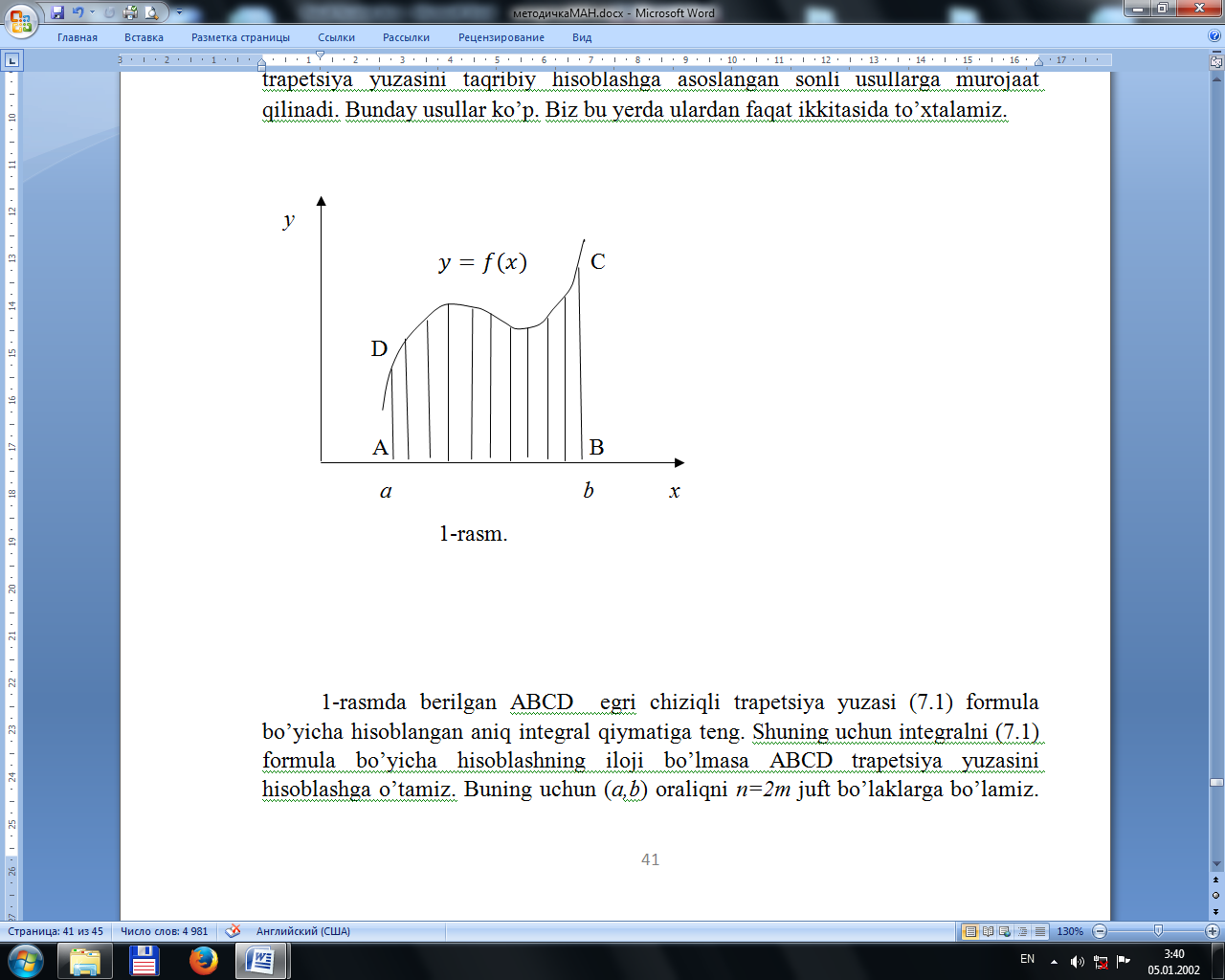
}

**1.1. Taqribiy integrallash usullari. Zaruriy aniqlikni ta’minlovchi qadamni tanlash.**

Ma’lumki, agar integral osti funksiyasi ning boshlang’ich funksiyasi ni topish mumkin bo’lmasa aniq integralni hisoblashda Nyuton-Leybnits formulasi

(1.1)

ni tadbiq qilib bo’lmaydi. Bunday hollarda (1.1) aniq integralning geometrik ma’nosi, ya’ni funksiya grafigi bilan chegaralangan egri chiziqli trapetsiya yuzasini taqribiy hisoblashga asoslangan sonli usullarga murojaat qilinadi. Bunday usullar ko’p. Biz bu yerda ulardan faqat ikkitasida to’xtalamiz.



1-rasm.

1-rasmda berilgan ABCD egri chiziqli trapetsiya yuzasi (1.1) formula bo’yicha hisoblangan aniq integral qiymatiga teng. Shuning uchun integralni (1.1) formula bo’yicha hisoblashning iloji bo’lmasa ABCD trapetsiya yuzasini hisoblashga o’tamiz. Buning uchun (*a,b*) oraliqni *n=2m* juft bo’laklarga bo’lamiz. Bo’linish nuqtalari

Simpson formulasiga ko’ra

(1.2)

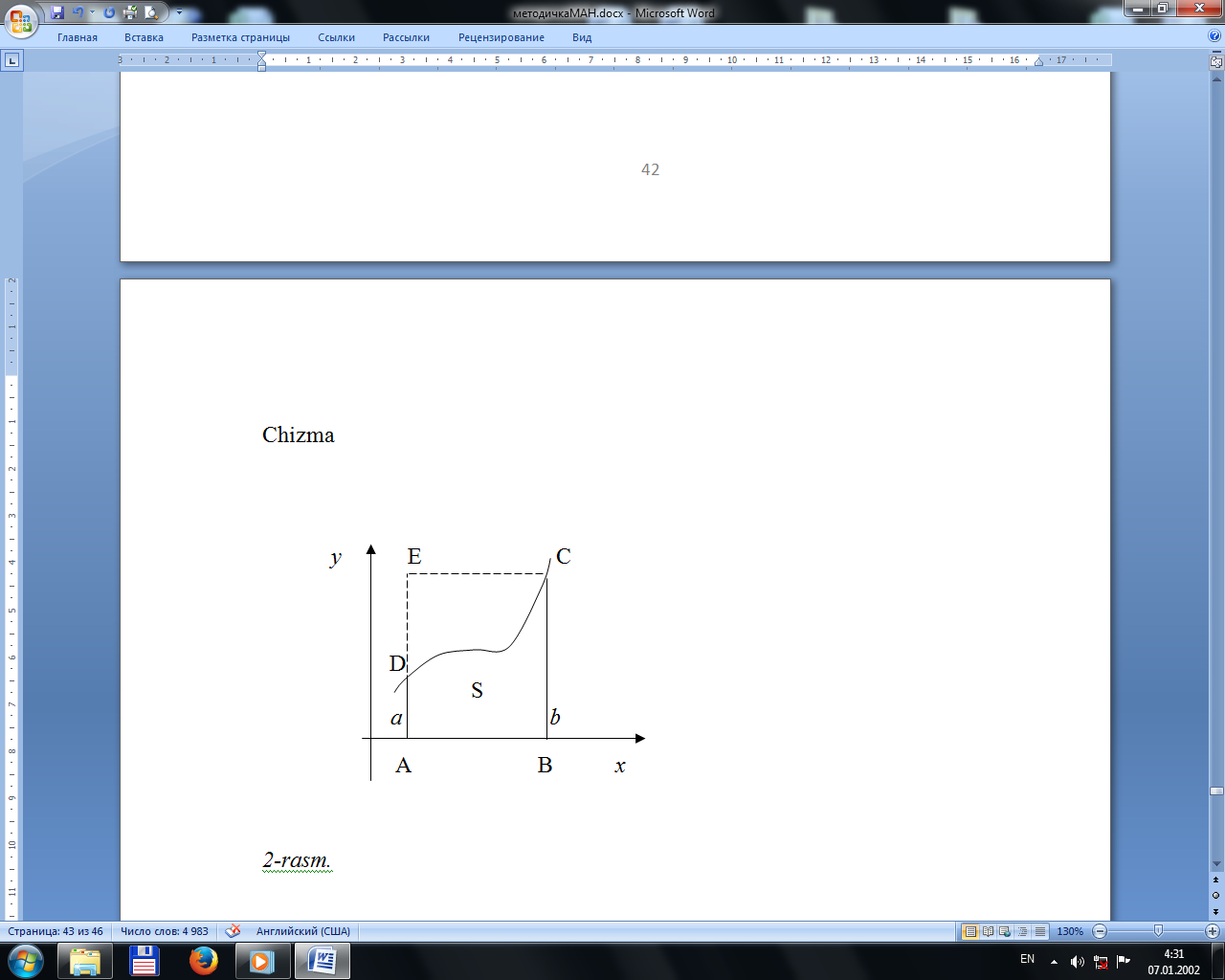
(1.2) formulani yoyib yozib yuborsak

(1.3)

formulani hosil qilamiz. (1.3) formula taqribiy formula bo’lib uning xatoligi tartibida bo’lar ekan. Bu degani, (1.3) Simpson formulasi sodda lekin ancha aniq formulalardan ekan. Amaliyotda bu formula juda keng qo’llaniladi. Uni dasturlash ham oson.

Monte-Karlo usuli esa ehtimolning geometrik va statistik ta’riflarini muvofiqlashtirishdan kelib chiqqan. Buning uchun *y=f(x)* funksiyani yuqori chegarasi topiladi.

Chizma

**

*2-rasm.*

2-rasmdagidek xolat o’rinli bo’lsin, ya’ni

U xolda

Ikkinchi tarafdan ehtimolning geometrik ta’rifiga ko’ra ABCE to’g’ri to’rtburchakka tavakkaliga tashlanadigan tasodifiy nuqta ABCD egri chiziqli trapetsiyaga tushish ehtimoli

(1.4)

Agar A – tasodifiy hodisa ABCE to’g’ri to’rtburchakka tavakkaliga tashlangan nuqtaning ABCD egri chiziqli trapetsiyaga tushishi deb qaralsa, bu xodisaning ehtimolini hisoblash uchun ehtimolning statistik ta’rifidan foydalanamiz. Buning uchun oraliqda tekis taqsimlangan tasodifiy miqdorlar va oraliqda tekis taqsimlangan tasodifiy miqdorlar ketma-ketligini tuzamiz. Buning uchun kompyuterda mavjud bo’lgan psevdotasodifiy miqdorlar generatoridan foydalanish mumkin. Hosil bo’lgan bu ketma-ketlikning har bir juftligi , ABCE to’g’ri to’rtburchakka taaluqli bo’ladi. Ulardan ABCD trapetsiyaga taaluqlilarini ajratamiz. Buning uchun shart bajarilishi kerak. Bunday nuqtalar soni *m* ta bo’lsin. U holda *A –* hodisa ehtimoli uchun

(1.5)

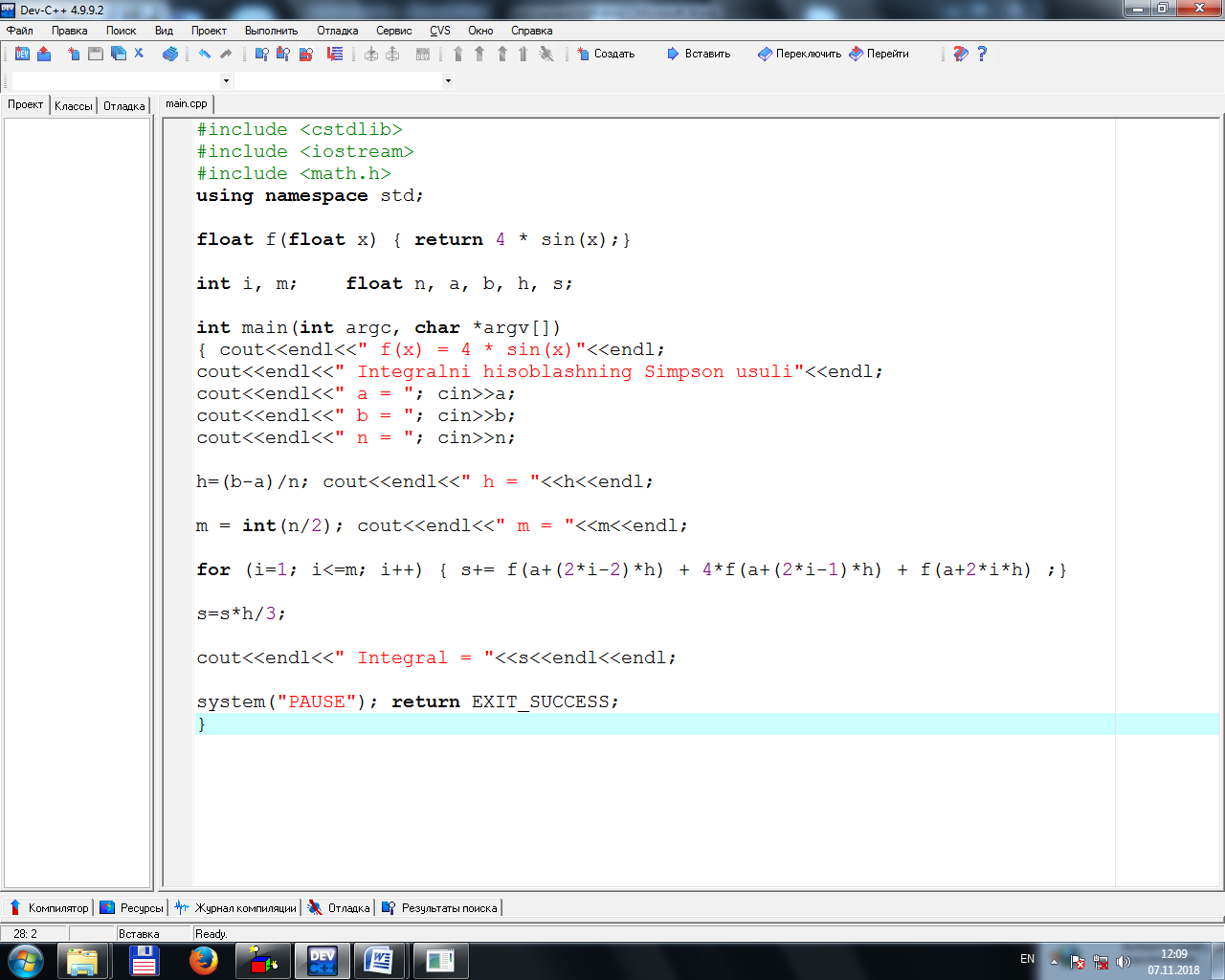
formuladan foydalanish mumkin. *n –* qanchalik katta bo’lsa (1.5) formula shunchalik aniq bo’ladi. (1.5) formuladan topilgan qiymatni (1.4) formulaga olib borib qo’yilsa

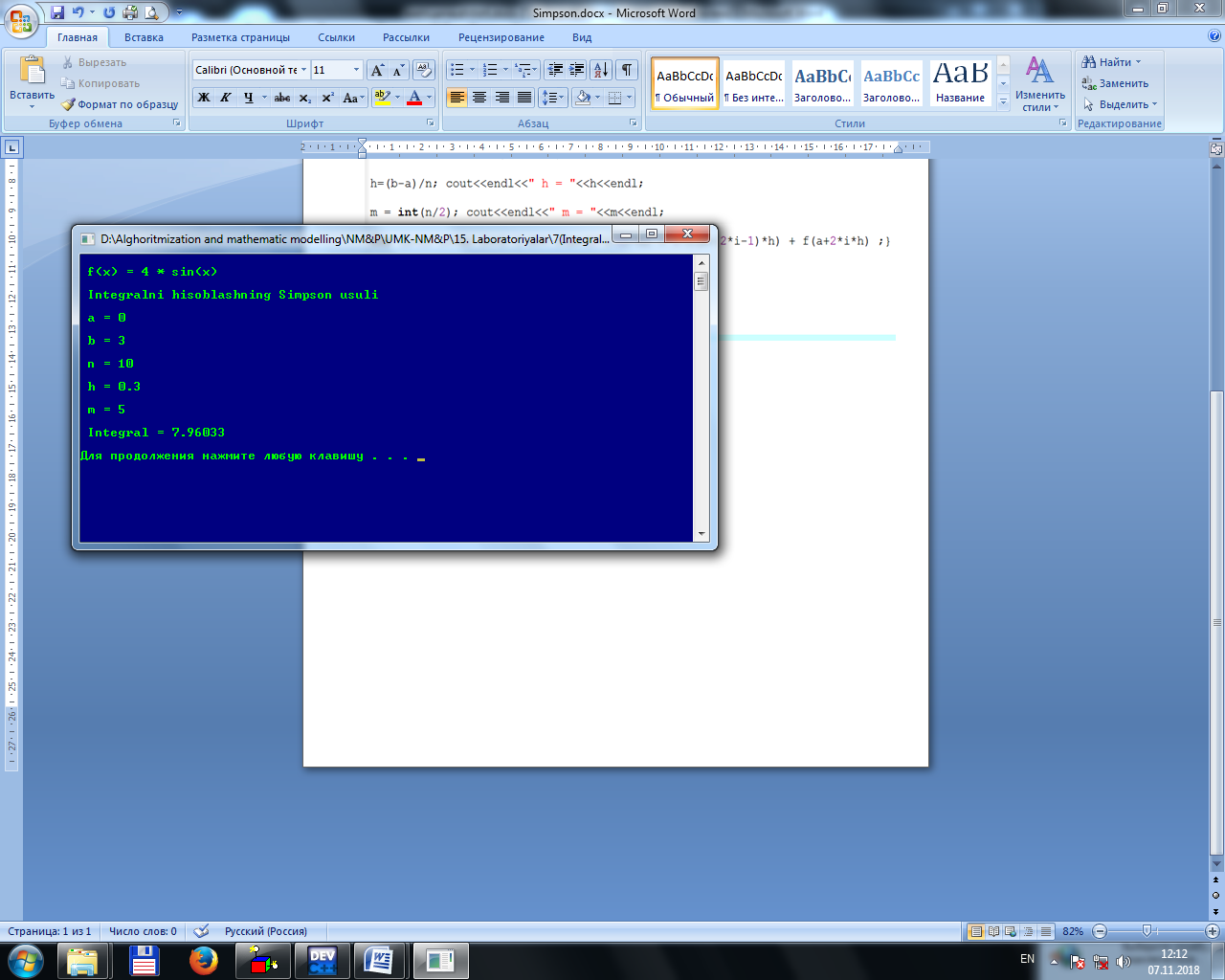
(1.6)

formula hosil bo’ladi. Integralni bu usulda hisoblash Monte-Karlo usuli deyiladi.

**Misol:** *f*(*x*) = 4 *sin*(*x*) funksiyani Simpson usulidan foydalanib [0; 3] oraliqda integrali hisoblansin.

**Yechilishi:** Quyida shu misolning C++ dastur kodi va natijasi keltirilgan.





**1.1-topshiriq varianlari**

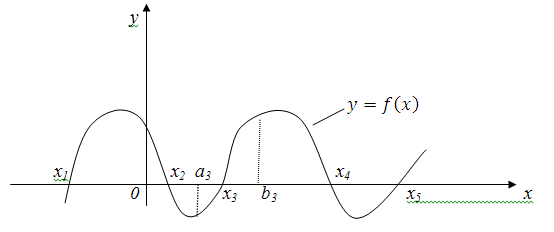
Berilgan integralni Simpson hamda Monte-Karlo usulida hisoblang. Oraliqni bo’linish soni *N* , hamda sinovlar soni *M* ko’rsatilgan.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** |  |  |  |  |
|  |  | [0;1] | 12 | 100 |
|  |  | [0;2] | 18 | 180 |
|  |  | [0,5;1,5] | 10 | 160 |
|  |  | [0;1] | 12 | 100 |
|  |  | [0;1] | 14 | 150 |
|  |  | [0;1] | 10 | 180 |
|  |  | [0;2] | 20 | 160 |
|  |  | [0;1] | 12 | 100 |
|  |  | [-1;1] | 20 | 150 |
|  |  | [0;1] | 10 | 140 |
|  |  | [0;1] | 10 | 140 |
|  |  | [1;2] | 12 | 100 |
|  |  | [0;2] | 20 | 150 |
|  |  | [0;1] | 10 | 110 |
|  |  | [1;2] | 12 | 200 |
|  |  | [0;1] | 10 | 120 |
|  |  | [0;2] | 18 | 200 |
|  |  | [1;3] | 16 | 150 |
|  |  | [0;1] | 10 | 100 |
|  |  | [1;3] | 16 | 120 |
|  |  | [1;3] | 20 | 140 |
|  |  | [0;2] | 16 | 200 |
|  |  | [0;1] | 10 | 160 |
|  |  | [0;1] | 10 | 100 |
|  |  | [1;3] | 12 | 120 |
|  |  | [0;2] | 14 | 150 |
|  |  | [0;2] | 18 | 200 |
|  |  | [0;0,5] | 10 | 100 |
|  |  | [0;2] | 20 | 150 |
|  |  | [1;2] | 12 | 100 |

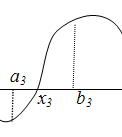
**1.2. Algebraik va transtsendent tenglamalarni yechishda oraliqni teng ikkiga bo’lish, iteratsiya usullari.**

Tenglamaning ildizi mavjudligi sharti:

Agar biror [*a,b*]oraliqda *y = f(x)* funksiya uzluksiz bo‘lib, *f(a) · f(b) < 0* bo‘lsa, shu oraliqda *f(x)=0* tenglamaning kamida bitta ildizi mavjud bo‘ladi.



Agar biror [*a,b*] oraliqda *y=f(x)* funksiya uzluksiz bo‘lib, birinchi tartibli uzluksiz xosilaga ega bo‘lsa va *f(a) · f(b) < 0 , f ‘ (x)* ([*a,b*] da ishorasi o‘zgarmasa) shartlar bajarilsa, *f(x)=0* tenglama shu oraliqda yagona xaqiqiy ildizga ega bo‘ladi.



Algebraik tenglamaning taqribiy yechimini berilgan [a;b] oraliqda topishni quyidagi algoritm bo’yicha tashkil qilamiz:

1. Berilgan [a;b] oraliqni o’rtasini hisoblaymiz.

 (1.7)

2. Yechimni [a;c] yoki [c;b] oraliqdaligini

*f(a)⋅ f(c)<0* (1.8)

shartidan foydalanib aniqlaymiz.

1. Shartni qanoatlantiradigan oraliqni yangi oraliq sifatida olamiz va uni teng ikkiga bo’lib, yuqoridagi amallarni yana takrorlaymiz.
2. Odatda tenglamaning taqribiy yechimini birorta aniqlik bilan topish so’raladi. Demak δ aniqlik berilgan bo’lsa, oraliqni bo’lish jarayonining xar bir qadamida ׀ b-a ׀ < δ (1.9) shart bajarilishi tekshiriladi. Shart bajarilganda oraliqning o’rta nuqtasi x\* , δ aniqlik bilan topilgan taqribiy yechim sifatida qabul qilinadi. Yangi oraliq uchun yuqoridagi ishlarni qayta takrorlaymiz va buni oraliq uzunligi δ -dan kichik bo’lmaguncha davom ettiramiz. Oxirgi oraliqni o’rta nuqtasini tenglamaning taqribiy yechimi sifatida qabul qilish mumkin.Oraliqni teng ikkiga bo’lish usulining algoritmi:

F (x) =…

с=(a+b)/2

*f(a)⋅f(с)<0*

b=c

a=c

b-a<*E*



с

a,b

F(a)⋅(f(b)≤0

Е

ха

йук

ха

ха

йук

йук

Oraliqni teng ikkiga bo’lish usuli uchun dastur kodi:

program ikkiga\_bolish;

var a,b,eps:real;

function f(x:real):real;

begin f:= {tenglamani o'ng tarafini yozing} end;

begin

write('a,b,eps=?');readln(a,b,eps);

1 : c:=(a+b)/2;

if f(a)\*f(c)<0 then b:=c else a:=c;

if (b-a)> e then goto 1;

write(' Yechim =' , (a+b)/2)

end.

**1-Vazifa.** Tenglamalar yechimlari joylashgan [a; b] oraliqni grafik va analitik usullar bilan ajrating.

**2-Vazifa.** Tenglamalar yechimlari joylashgan oraliqlar aniqlangandan so’ng taqribiy yechimlarini oraliqni teng ikkiga bo’lish usulida E=0.001 aniqlikda hisoblang. Algoritmini tuzib, dasturlash tilida dastur kodini yozib natija oling.

## **3-Vazifa**. Algebraik va transtsendent tenglamalarning taqribiy yechimlarini vatarlar va urinmalar usuli bilan toping. Algoritmini tuzib, dasturlash tilida dastur kodini yozib natija oling.

## **Laboratoriya ishiga doir topshiriq variantlari:**

1. a) 2x­­­­­­­3-2x-1=0 b) 3x+cosx+1=0

2. a) x­­­­­­­3-x+7=0 b) lnx+2=0

3. a) 2x­­­­­­­3-2x2+3x+1=0 b) x+cosx-1=0

4. a) 2x­­­­­­­3-x-5=0 b) 

5. a) x­­­­­­­3-3x2+2x-4=0 b) x2+4⋅sinx=0

6. a) x­­­­­­­3+2x2+5x+2=0 b) lnx+x+1=0

7. a) 2x­­­­­­­3+2x-4=0 b) 2x-lgx=3

8. a) x­­­­­­­3-2x2+7x-1=0 b) 

9. a) 2x­­­­­­­3+3x+4=0 b) x2=3sinx

10. a) x­­­­­­­3-3x2+6x+2=0 b) 3x-2lnx=4

11. a) x­­­­­­­3-2x+2=0 b) 4x-ex=0

12. a) x­­­­­­­3-3x2+2x-4=0 b) x⋅(x+1)2=2

13. a) x­­­­­­­3+x-8=0 b) 3-2x=lnx

14. a) x­­­­­­­3-3x2+5x+1=0 b) 2x-cosx=0

15. a) x­­­­­­­3-x+2=0 b) sin(x/2)+1=x2

16. a) x­­­­­­­3-3x2+7x+1=0 b) 2x+lgx=-0,5

17. a) x­­­­­­­3-3x+1=0 b) (2-x)⋅ex=1

18. a) x­­­­­­­3+x2+2x+4=0 b) x3=2sinx

19. a) x­­­­­­­3-2x-5=0 b) 2x-2x=0

20. a) x­­­­­­­3+2x2+3x-2=0 b) x2-4⋅sinx=0

21. a) x­­­­­­­3+4x-6=0 b) x2=ln(x+2)

22. a) x­­­­­­­3-3x2+6x-5=0 b) 2x-cosx=0

23. a) x­­­­­­­3-2x+7=0 b) 3x+cosx=2

24. a) x­­­­­­­3-4x+1=0 b) x+lgx=1,5

25. a) x­­­­­­­3+2x+1=0 b) x-3=0

26. a) 2x­­­­­­­3-3x-5=0 b) 2x-2x-1=0

27. a) x­­­­­­­3+(1/2)x-2=0 b) x2-sosx=0

28. a) x­­­­­­­3-x+1=0 b) (2+x)⋅ex=1

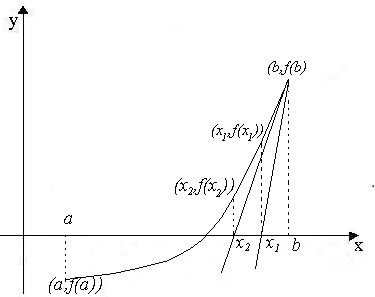
29. a) x­­­­­­­3+x2+x+4=0 b) x3=2cos(x)

30. a) x­­­­­­­3-3x+15=0 b) 2x-ex=0

**Tenglamalarni yechishda Nyuton va vatarlar usullari. Yaqinlashish tezligi.**

**Nyuton (Urinmalar)** **usuli.**

f(x)=0 tenglama berilgan. Biror [a;b] oraliqda f(a)\*f(b)<0 bo’lsin. [a,b] oraliqdagi (b,f(b)) nuqtadan urinma o’tkazamiz.





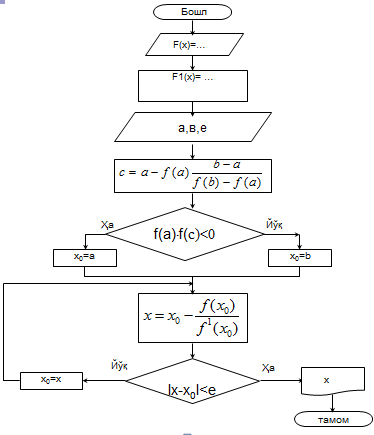
 



 (1.10)  (1.11)

Nyuton (Urinma) usuli yordamida [a;b] oraliqda  aniqlida taqribiy ildizlarini topish algoritm blok sxemasi.



**Misol**. tenglama taqribiy yechimini ε=0.01 aniqlik bilan toping.

**Yechish.**  funktsiya [-1;0] oraliqda 1.3-teoremaning barcha shartlarini qanoatlantiradi.

, ­x∈[-1;0] va *f*(-1)=8.386>0 dan

bo’lgani uchun *a*0=-1 deb olinadi. ni e’tiborga olib, birinchi yaqinlashish *a1* ni hisoblaymiz:

Yaqinlashish shartini tekshiramiz:

­­| *a*1- *a*0 | = |-0.131+1|= 0.869>ε=0.01

bo’lgani uchun ikkinchi yaqinlashish *a*2 ni

formula bilan topamiz.

lar asosida: *a*2=-0.131- 0.1895/(-9.123) = -0.1104.

Yana |*a2- a1*| *=* 0.0214 > ε bo’lgani uchun *a3* ni topamiz.

lar asosida:

yaqinlashish sharti |*a*3-*a*2|< ε=0.01 bajarilganligi uchun tenglamaning ε=0.01 aniqlikdagi taqribiy yechimi:

x *a*3= -0.11 bo’ladi.

**Urinmalar usuli uchun dastur kodi:**

Program Urinma;

Label 1,2,3,4;

Var a,b,x1, x2, eps : real;

Function F (x: real): real; Begin F: = … end;

Function F 1(x: real): real; Begin F 1: = … end;

Function F 2(x: real): real; Begin F 2: = … end;

Begin

writeln(‘a,b=’); readln(a,b);

writeln(‘ aniqlikni kiriting'); readln( eps);

if F1(a)\*F2(a)>0 then x1:=b else goto 2;

1: x2:=x1 – F(x1) / F1(x1);

If abs(x2-x1)>eps then begin x1:=x2;goto1 end else goto3;

2 : if F1(a)\*F2(a)<0 then x1:=a;

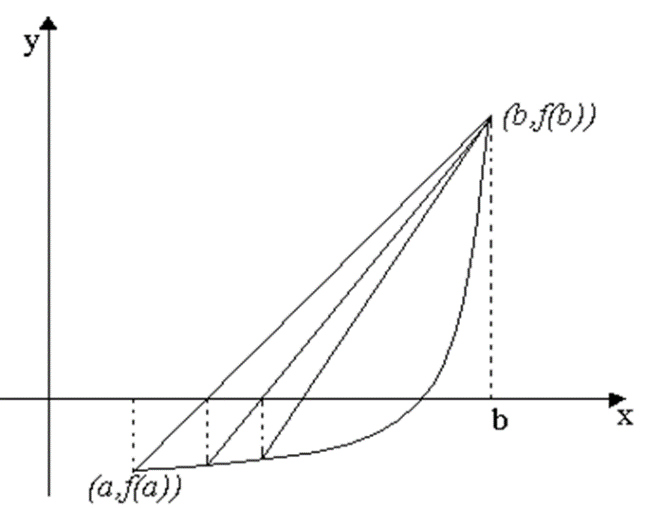
4: x2:=x1 – F(x1) / F1(x1);

If abs(x2-x1)>eps then begin x1:=x2;goto 4 end ;

3 : Writeln (‘tenglama yechimi= ‘,x); End.

**Vatarlar usuli**

f(x)=0 tenglama berilgan. Biror [a;b] oraliqda f(a)\*f(b)<0 bo’lsin. [a;b] oraliqdagi (a,f(a)) va (b,f(b)) nuqtalardan vatar o’tkazamiz.



|  |  |
| --- | --- |
|  |  |
|  |  |
|  | (1.11) |
|  | (1.12) |

**Misol**. tenglamaning ε =0. 01 aniqlikdagi taqribiy ildizi topilsin.

**Yechish**. Ma’lumki funksiya [-1;0] oraliqda teoremalarning hamma shartlarini bajaradi. ­x∈[-1;0] da ikkinchi tartibli hosila . Demak f(0)=-1, f(-1)=8.368 bo`lganligi uchun, f(a)\*f(b)<0 shartga asosan f(0)f''(0)<0 bo`lgani uchun {*a*n} ketma-ketlik vatarni topish formulasi bilan topiladi.

Berilganlar: *a*=-1, *b*=0, ε=0. 01

f(x)= yex-10x-2, f(-1)=e-1 -10(-1) -2=8. 386, f(0)=e0-10\*0-2=-1

vatar ildizlarini topish formulasiga asosan:

*b*0= *0*

*b 1= b0* - (*a- b*0) f(*b0*)/ (f(*a*)-f(*b0*))= -0.107

Yaqinlashish sharti ⏐ *b1 - b2*⏐>ε bo`lganligi uchun *b2* yaqinlashishni hisoblaymiz. Buning uchun

*b1=* -0.107, f(-0.107)=e-0.107-10(-0.107)-2 =-0.038 , f(a)=f(-1)=8.386

larga asosan:

*b2*= *b1* - (*a- b 1*) f(*b 1*)/ (f(*a*)-f(*b 1*)) = 0.111

⏐ *b2- b1*⏐+⏐- 0.111+0.107⏐=0.004<ε=0. 01

Demak taqribiy yechim deb t= *bn* =-0.111 ni olish mumkin.

**Vatarlar usuli uchun dastur kodi:**

Program Vatar;

Label 1,2,3,4;

Var a,b,x1, x2, eps : real;

Function F (x: real): real; Begin F: = … end;

Function F 1(x: real): real; Begin F 1: = … end;

Function F 2(x: real): real; Begin F 2: = … end;

Begin

writeln(‘a,b=’); readln(a,b);

writeln(‘ aniqlikni kiriting'); readln( eps);

if F1(a)\*F2(a)>0 then x1:=a else goto 2;

1: x2:=x1 – F(x1)\*(b-x1) /(F(b)-F(x1));

If abs(x2-x1)>eps then begin x1:=x2;goto 1 end else goto 3;

2 : if F1(a)\*F2(a)<0 then x1:=b;

4: x2:=x1 – F(x1)\*(x1-a) / (F(x1)-F(a));

If abs(x2-x1)>eps then begin x1:=x2;goto 4 end ;

3 : Writeln (‘tenglama yechimi= ‘,x);

End.

Dastur kodida: F - tenglamani o’ng tomoni;

F1 - tenglama o’ng tomonidan olingan birinchi hosila

F2 - tenglama o’ng tomonidan olingan ikkinchi hosila

a , b – oraliqni chap va o’ng chegaralari.

Eps – hisoblash aniqligi.

**Talabalar mustaqil bajarishlari uchun topshiriqlar**.

1) Tenglama ildizlarini ajratish iteratsion metodi yordamida 0,001 aniqlikda hisoblash.

2) Vatarlar va urinmalar usullari yordamida tenglama taqribiy ildizlarini 0,001 aniqlikda hisoblash.

3) Tenglamaning taqribiy oraliqlari topib, ularning yaqinlashish tezligini baholash.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |