**Отчет ПО ЛАБОРАТОРНЫМ РАБОТАМ**

**По информатике**

Лабораторная работа №3.

Языки программирования

(наименование лабораторной работы в соответствии с учебным планом)

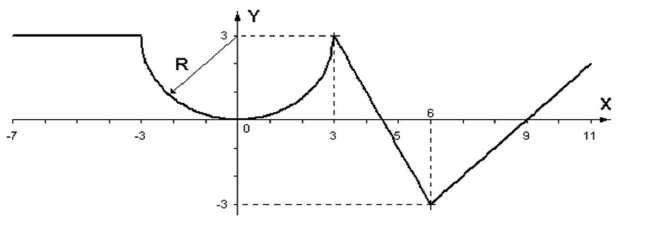
|  |  |
| --- | --- |
|  |  |

***Цель и задачи работы:***

Изучение циклических алгоритмов, операторов цикла, программирование циклического вычислительного процесса.

***Задание №1***

Вычислить и вывести на экран или в файл в виде таблицы значения функции, заданной графически, на интервале от *Xнач* до *Xкон* с шагом *dx*. Интервал и шаг задать таким образом, чтобы проверить все ветви программы. Таблица должна иметь заголовок и шапку.



***Алгоритм решения:***

Блок-схема алгоритма решения задачи представлена ниже.

Начало

xmin = -7

xmax = 11

x = xmin

R = 3

Ввод dx

нет

x = x + dx

x <= xmax

да

Вывод контейнера map

x <= -3

да

mp[x] = 3

Конец

нет

-3 < x <= 3

да

mp[x] =

нет

3 < x <= 6

да

mp[x] = 9 - 2x

нет

6 < x <= 1

да

mp[x] = x - 9

***Листинг программы:***

1) C++

#include <iostream>

#include <map>

#include <cmath>

#include <iomanip>

using namespace std;

void printmap(map <float, float> m) {

for (auto i : m) cout << setw(6)<< i.first << setw(38) << i.second << endl;

}

int main()

{

int xmin = -7;

int xmax = 11;

float x;

float dx;

cout << "Enter dx: ";

cin >> dx;

int R = 3;

map <float, float> mp;

cout << setw(41)<< "\nvalues of function with step dx = " << dx << endl << endl;

cout << setw(6)<<"X" << setw(39) << "F(x)" << endl << endl;

for (x = xmin; x <= xmax; x += dx) {

if (x <= -3) mp[x] = 3;

if ((x > -3) && (x <= 3)) mp[x] = (6.0 - sqrt(36.0 - (4.0 \* x \* x))) / 2.0;

if ((x > 3) && (x <= 6)) mp[x] = 9 - (2 \* x);

if ((x > 6) && (x <= 11)) mp[x] = x - 9;

}

printmap(mp);

}

2) Python

import math

x = -7

dx = 1

i = 0

values = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18]

print("\t\t Table of function value calculations ")

print("Step", "\t\t Argument (x)", "\t\tFunction value")

for element in values:

if x <= 3 : values[i] = 3;

if -3 < x <= 3: values[i] = (6.0 - math.sqrt(36.0 - (4.0 \* x \* x)))/2.0

if 3 < x <= 6: values[i] = 9 - (2 \* x)

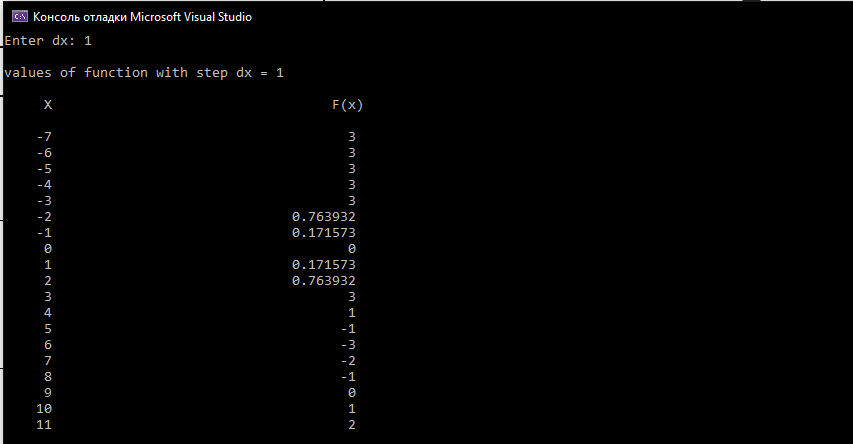
if 6 < x <= 11: values[i] = x - 9;

print (dx, "\t\t\t", x, "\t\t\t", values[i])

i += 1

x += 1

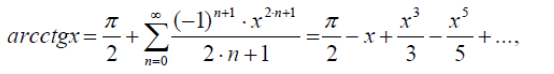
***Пример работы программы:***



*(рис. 1 – пример работы программы на C++)*

**Задание №2.**

Вычислить и вывести на экран в виде таблицы значения функции, заданной с помощью степенного ряда, на интервале от *Xнач* до *Xкон* с шагом *dx* с точностью ε (вводится пользователем). Таблица должна иметь заголовок и шапку. Каждая строка таблицы должна содержать значение аргумента, значение функции и количество просуммированных членов ряда.



***Алгоритм решения:***

Блок-схема алгоритма решения задачи представлена ниже.

Начало

xmin = -1

xmax = 1

dx = 0.25

pi = 3.14

ввод N

x <= xmax

нет

x = x + dx

да

Вывод контейнера map

mp[x] = summ

summ = 0

нет

summ = summ + pi/2

n <= 100

Конец

да

<= N

да

нет

summ = summ +

n = n +1

***Листинг программы:***

1) C++

#include <iostream>

#include <cmath>

#include <iomanip>

#include <map>

#include <stdio.h>

using namespace std;

void Printmap(map <float, float> m) {

for (auto i : m) cout << setprecision(2) << i.first << setw(10) << fixed << i.second << endl;

}

int main()

{

float xmin = -1;

float xmax = 1;

float dx = 0.25;

const float pi = 3.14;

map <float, float> mp;

int n;

cout <<"Table of function value calculations: arcctgx ( -1 <= x <= 1)" << endl;

cout << "Enter the accuracy: ";

float N;

cin >> N;

cout << endl;

float x;

cout << setw(25) << "values of function with step = 0.25\n";

cout<<endl;

cout << setw(2) << "X" << setw(12) << "F(x)" << endl;

float summ;

for (x = xmin; x <= xmax; x += dx) {

summ = 0;

for (n = 0; n <= 100; n++) {

if (abs(((pow((-1), (n + 1)) \* pow(x, ((2.0 \* n) + 1))) / ((2.0 \* n) + 1))) <= N) break;

summ = summ + ((pow((-1), (n + 1)) \* pow(x, ((2.0 \* n) + 1))) / ((2.0 \* n) + 1));

}

summ = summ + (pi / 2);

mp[x] = summ;

}

Printmap(mp);

}

2) Python

import math

PI = 3.14

dx = 0.25

x = -1.0

n = 0

i = 0

values = [0, 1, 2, 3, 4, 5, 6, 7, 8]

print ("\t\t Table of function value calculations: arcctgx ( -1 <= x <= 1)")

print ("Enter the accuracy: ")

N = int(input())

print("\n")

print("Step(dx)", "\t\tX", "\t\t F(x)", "\t\tAccuracy")

for element in values:

summ = 0

for n in range(N + 1):

summ = summ + ((pow((-1), (n + 1)) \* pow(x, ((2.0 \* n) + 1))) / ((2.0 \* n) + 1))

summ = summ + (PI / 2)

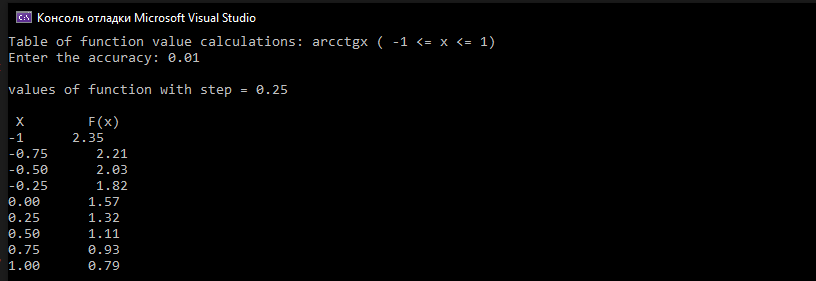
values[i] = round (summ, 2)

print(dx, "\t\t ", x, "\t\t", values[i], "\t\t ", n)

i += 1

x += dx

***Пример работы программы:***



*(рис. 2 – пример работы программы на C++)*

**Задание №3:**

Реализовать ax mod p Сравнения по модулю простого числа через теорему Ферма и свойства сравнений.

***Алгоритм решения:***

Блок-схема алгоритма решения задачи представлена ниже.

начало

Ввод number

Ввод degree

Ввод prime number

result = 1

i = 1

degree = degree % (prime -1)

number = number % p

i <= degree

нет

Вывод result

да

result = result \* a

result = result %p

Конец

i = i +1

***Листинг программы:***

1)C++

#include <iostream>

using namespace std;

int main()

{

int a, b, p;

cout << "Enter the number: ";

cin >> a;

cout << "\n" << "Enter degree: ";

cin >> b;

cout << "\n" << "Enter a prime number: ";

cin >> p;

b = b % (p - 1);

a = a % p;

int result = 1;

for (int i = 1; i <= b; i++) {

result \*= a;

result = result % p;

}

cout <<"\nAnswer: a ^ b mod p = "<< result;

}

2) Python

print("The program calculates: a^b mod p")

print ("Enter the number: ")

a = int(input())

print("Enter degree: ")

b = int(input())

print("Enter a prime number")

p = int(input())

b = b % (p - 1)

a = a % p

result = 1

i = 1

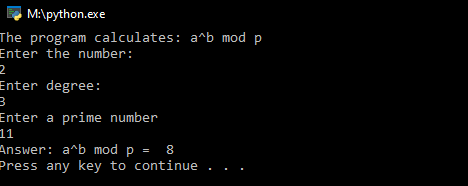
for i in range(b):

result \*=a

result = result % p

print("Answer: a^b mod p = ", result)

***Пример работы программы:***



(*рис. 3 – пример работы программы на Python*)

**Задание №4.**

Реализовать обобщенный алгоритм Евклида для вычисления с\*d mod m=1.

***Алгоритм решения:***

Блок-схема алгоритма решения задачи представлена ниже.

Начало

Вывод z

z < 0

Q != 0

Вывод vector X

Вывод vector Y

Ввод a

Ввод b

Vector X = {a,1,0}

Vector Y = {b,0,1}

Q =1

q = 1

c = 0

да

да

да

нет

нет

нет

Конец

Вывод vector T

vector T = { X[0] % Y[0], X[1] - (q \* Y[1]), X[2] - (q \* Y[2])}

c = T[2]

X = Y

Y = T

z = c

Q == 0

z = z + a

Q = X[0] % Y[0]

q = X[0] / Y[0]

***Листинг программы:***

1)C++

#include <iostream>

#include <vector>

#include <iomanip>

using namespace std;

void Print(vector <int> v) {

for (auto i : v) cout << setw(10) << i << setw(10) ;

}

void fn1(int a, int b)

{

vector<int> X = { a, 1, 0 };

vector<int> Y = { b, 0, 1 };

int Q = 1;

int q = 1;

int c = 0;

Print(X);

cout << endl;

Print(Y);

int z;

cout << endl;

while (Q != 0)

{

Q = X[0] % Y[0];

q = X[0] / Y[0];

if (Q == 0) z = c;

vector <int> T = { X[0] % Y[0], X[1] - (q \* Y[1]), X[2] - (q \* Y[2])};

c = T[2];

Print(T);

cout << setw(5) << " | " << setw(5) << q << endl;

X = Y;

Y = T;

}

if (z < 0) z += a;

cout << "\n\n" << setw(5) << "Answer : " <<b << "^(-1) mod " << a << " = " << z << endl;

}

int main()

{

cout << "The program finds the value of the expression: z = b^(-1) mod a " << endl;

int a, b;

cout << "Enter a: ";

cin >> a;

cout << "\nEnter b: ";

cin >> b;

cout << endl << setw(13) << "Remains" << setw(7) << "X" << setw(11) << " Y " << setw(10) << " q "<<endl;

cout << "----------------------------------------" << endl;

fn1(a, b);

}

2) Python

import math

def printed(vector, q):

print (vector[0], "\t\t\t" , vector[1],"\t\t", vector[2], "\t\t", q)

def fn1(a,b):

X = [a,1,0]

Y = [b,0,1]

Q = 1

q = 0

c = 0

printed(X,q)

printed(Y,q)

q = 1

while Q != 0:

Q = X[0] % Y[0]

q = math.floor(X[0] / Y[0])

if Q == 0:

z = c

T = [X[0] % Y[0], X[1] - (q \* Y[1]), X[2] - (q \* Y[2])]

c = T[2]

printed(T,q)

X = Y

Y = T

if z < 0 :

z += a

print("\n\t Answer:", b, "^(-1) mod ", a, " = ", z)

print("The program finds the value of the expression: z = b^(-1) mod a")

print("Enter a: ")

a = int(input())

print("Enter b: ")

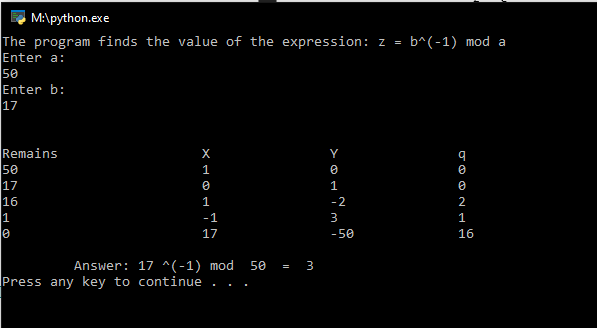
b = int(input())

print("\n")

print("Remains", "\t\t X","\t\t Y", "\t\t q")

fn1(a,b)

***Пример работы программы:***



(*рис. 4 – пример работы программы на Python*)

**Задание №5.**

Написать программу, использующую алгоритм шифрования данных для преобразования исходного текста. Шифрование Шамира.

***Алгоритм решения:***

Блок-схема алгоритма решения задачи представлена ниже.

**Блок-схемы отдельных функций:**

1) mod (number, degree, p):

Начало

number

degree

prime number

result = 1

i = 1

degree = degree % (prime -1)

number = number % p

i <= degree

нет

Вывод result

да

result = result \* a

result = result %p

Конец

i = i +1

2) Evklid (a , b):

Начало

Ввод a

Ввод b

Vector X = {a,1,0}

Vector Y = {b,0,1}

Q =1

q = 1

c = 0

vector T = { X[0] % Y[0], X[1] - (q \* Y[1]), X[2] - (q \* Y[2])}

c = T[2]

X = Y

Y = T

нет

z = c

да

Q == 0

Q = X[0] % Y[0]

q = X[0] / Y[0]

Конец

return z

z = z + a

да

нет

z < 0

нет

да

Q != 0

3) encrypt (message, m):

Начало

i = 0

message[i] != 0

return message

Конец

message[i] = (char) (message[i] -m)

4) decrypt (message, k):

Начало

i = 0

message[i] != 0

return message

Конец

message[i] = (char) (message[i] + k)

**Общая блок-схема:**

Начало

Конец

Вывод decrypted

first = mod(m, cA, p)

second = mod(first, cB, p)

third = mod(second, dA, p)

fourth = mod(third, dB, p)

decrypted = decrypt(encrypted, fourth)

Вывод encrypted

dA = Evklid(p-1, cA)

dB = Evklid(p-1, cB)

encrypted = encrypt(message, m)

Ввод простого числа (p)

Ввод открытого ключа для А (cA)

Ввод открытого ключа для В (cB)

Ввод секретного номера (m)

Ввод сообщения (message)

***Листинг программы:***

1) C++

#include <iostream>

#include <vector>

using namespace std;

int mod(int number, int degree, int p) {

degree = degree % (p - 1);

number = number % p;

int result = 1;

for (int i = 1; i <= degree; i++) {

result \*= number;

result = result % p;

}

return result;

}

int Evklid(int a, int b) {

vector<int> X = { a, 0 };

vector<int> Y = { b, 1 };

int Q = 1;

int q = 1;

int c = 0;

int z;

while (Q != 0)

{

Q = X[0] % Y[0];

q = X[0] / Y[0];

if (Q == 0) z = c;

vector <int> T = { X[0] % Y[0], X[1] - q \* Y[1] };

c = T[1];

X = Y;

Y = T;

}

if (z < 0) z += a;

return z;

}

string encrypt(string message, int m) {

for (int i = 0; message[i] != 0; i++) {

message[i] = (char)(message[i]-m);

}

return message;

}

string decrypt(string message, int k) {

for (int i = 0; message[i] != 0; i++) {

message[i] = (char)(message[i] + k);

}

return message;

}

int main()

{

cout << "Enter a prime number: ";

int p;

cin >> p;

cout << "\nEnter key for user A: ";

int cA;

cin >> cA;

int dA = Evklid(p - 1, cA);

cout << "\nEnter key for user B: ";

int cB;

cin >> cB;

int dB = Evklid(p - 1, cB);

cout << "\nEnter secret base number: ";

int m;

cin >> m;

cout << "\nEnter the message you want to encrypt: ";

string message;

cin >> message;

string encrypted = encrypt(message, m);

cout << "\nEncrypted message: " << encrypted << endl;

int first = mod(m, cA, p); // A -> B

cout << "\n A --> B : " << first << endl;

int second = mod(first, cB, p); // B -> A

cout << "B --> A : " << second << endl;

int third = mod(second, dA, p); // A -> B

cout << "A --> B : " << third << endl;

int fourth = mod(third, dB, p); // B decrypted

cout << "B decrypted and get: " << fourth << endl;

cout << "\nIn theory, the value deciphered by B should coincide with m, lets chek it: " << fourth << " == " << m << endl;

cout << "\n Decrypted message: " << decrypt(encrypted, fourth) << endl;

}

2) Python

def Evklid(a,b):

X = [a,0]

Y = [b,1]

Q = 1

q = 0

c = 0

q = 1

while Q != 0:

Q = X[0] % Y[0]

q = round(X[0] / Y[0], 0)

if Q == 0:

z = c

T = [X[0] % Y[0], X[1] - (q \* Y[1])]

c = T[1]

X = Y

Y = T

if z < 0 :

z += a

return z

def encrypt(message, m):

z = ""

for i in message:

x = ord(i) - m

z += chr(x)

return z

def mod(number, degree, p):

degree = round(degree % (p - 1))

number = round(number % p)

result = 1

i = 1

for i in range(degree):

result \*=number

result = result % p

return result

def decrypt(message, m):

z = ""

for i in message:

x = ord(i) + m

z += chr(x)

return z

print("Enter a prime number:")

p = int(input())

print("Enter key for user A:")

cA = int(input())

dA = Evklid(p-1, cA)

print("Enter key for user B:")

cB = int(input())

dB = Evklid(p-1, cB)

print("Enter secret base number: ")

m = int(input())

print ("Enter the message you want to encrypt: ")

message = input()

encrypted = encrypt(message, m)

print("\nEncrypted message: ", encrypted)

first = mod(m,cA,p)

second = mod(first,cB,p)

third = mod(second,dA,p)

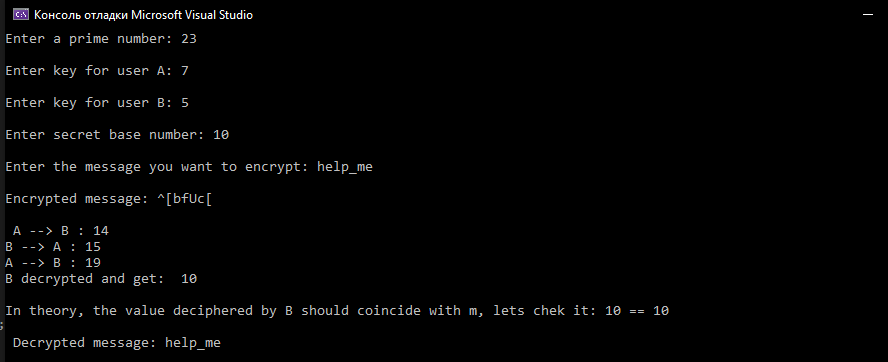
fourth = mod(third,dB,p)

print ("\nA -> B: ", first, "\nB -> A: ", second, "\nA -> B: ", third, "\nB decrypted and get: ", fourth)

print("In theory, the value deciphered by B should coincide with m, lets chek it: ", fourth, " == ", m)

print("\nDecrypted message: ", decrypt(encrypted, fourth))

***Пример работы программы:***



(*рис. 5 – пример работы программы наC++*)