МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

### федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

(ГУАП)

КАФЕДРА ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ И СЕТЕЙ

Преподаватель

канд. техн. наук, доцент Л.Н. Бариков

Отчёт

по лабораторной работе №4

по дисциплине ОСНОВЫ ПРОГРАММИРОВАНИЯ

на тему: «Операторы цикла: вычисления с заданной точностью»

Работу выполнил

студент гр. 4941 Н.С. Горбунов

Санкт-Петербург

2020

***Цель лабораторной работы:*** *изучение концепций и освоение технологии процедурного программирования, приобретение навыков процедурного программирования на языке C/C++ циклических вычислений.*

***Задание на программирование:*** *используя технологию процедурного программирования разработать программу решения индивидуальной задачи тремя видами циклических управляющих структур: Цикл - Пока (с предусловием), Цикл - До (с постусловием), Цикл - Для (с параметром).*

***Порядок выполнения работы:***

1. Получить у преподавателя индивидуальное задание. Выполнить постановку задачи: сформулировать условие, определить входные и выходные данные.

2. Разработать математическую модель.

3. Построить схемы алгоритмов трёх функций, использующих для решения задачи циклические управляющие структуры с операторами ***while***, ***do…while***, ***for*** соответственно.

При этом **запрещается** использовать оператор ***if*** для проверки условия входа в циклы и в качестве одного из операторов тела цикла для проверки условия выхода из цикла.

**Вывести результат и число итераций.**

4. Составить программу на языке *C/C++*.

5. Входные данныеоднократновводятся с клавиатуры по запросу. При этом под точностью вычисления результата понимается величина *ε*, значение которой лежит в диапазоне (0, 0.1].

6. Ввод исходных данных и вывод результатов должен осуществляться в функции ***main***().

Обеспечить контроль правильности ввода значения точности.

Контролировать выход числа итераций за значение INT\_MAX.

Выходные данные выводить на экран в развернутой форме с пояснениями.

7. Использовать стандартные потоковые объекты ввода/вывода ***cin*** и ***cout***.

8. Проверить и продемонстрировать преподавателю работу программы на полном наборе тестов.

9. Оформить отчет о лабораторной работе в составе: постановка задачи, математическая модель, схемы алгоритмов трёх функций, текст программы, контрольные примеры (скриншоты).

**Задание 13**

Вычислить предел последовательности {*Yn*} при *n*→, где *Yn* вычисляется по формуле:



Значения *Y*0, *Y*1, *Y*2 и точность вычисления *ε* вводятся с клавиатуры. Вычисления прекратить при выполнении условия ⎪*Yn*-*Yn*-1⎪≤*ε*.

**Математическая модель**

**Для цикла с предусловием:** При вызове в функцию ***Before\_cond*** передаются значения *ε*, *Y*0, *Y*1, *Y*2 и указатель на параметр *n*. На основе этих значений для значения параметра *n*=3 вычисляется значение *Y* по заданной рекуррентной формуле. Проверяется выполнение условия прекращения вычислений ⎪*Y*–*Y*2⎪≤*ε*. Для контроля числа выполненных итераций одновременно проверяется выполнение условия *n*<INT\_MAX.

Если ⎪*Y*–*Y*2⎪>*ε*, для вычисления следующего значения *Y* увеличивается на 1 значение параметра *n*, значение *Y*1 переписывается в *Y*0, значение *Y*2 переписывается в *Y*1, значение *Y* переписывается в *Y*2 и снова вычисляется значение *Y*. Действия повторяются до выполнения условия ⎪*Y*–*Y*2⎪≤*ε*. Необходимо обратить внимание, что очередное значение *Y* **всегда вычисляется непосредственно перед проверкой** выполнения условия прекращения вычислений.

При завершении работы функция возвращает значение *Y*. Значение параметра *n* возвращается через указатель.

**Для цикла с постусловием:** Необходимо обратить внимание, что поскольку очередное значение *Y* всегда вычисляется **непосредственно перед проверкой** выполнения условия прекращения вычислений, построение схемы алгоритма с использованием цикла с постусловием **целесообразно вести** **снизу вверх** от проверки выполнения условия прекращения вычислений ⎪*Y*–*Y*2⎪≤*ε*.

Перед этой проверкой вычисляется очередное значение *Y* по заданной рекуррентной формуле. Перед этим предыдущее значение *Y* переписывается в *Y*2, перед этим значение *Y*2 переписывается в *Y*1, перед этим значение *Y*1 переписывается в *Y*0, перед этим увеличивается на 1 значение параметра *n*.

Поэтому для правильного выполнения цикла **параметр *n* при вызове функции** (т.е. перед входом в цикл) **принимает значение 2**, а в функцию должны быть переданы значения *ε*, *Y*0, *Y*1, *Y*2 и указатель на параметр *n*. При вызове функции значения *Y*0, *Y*1, *Y*2 присваиваются соответственно переменным *Y*1, *Y*2, *Y*.

Действия повторяются до выполнения условия ⎪*Y*–*Y*2⎪≤*ε*. Для контроля числа выполненных итераций проверяется выполнение условия *n*<INT\_MAX.

При завершении работы функция возвращает значение *Y*. Значение параметра *n* возвращается через указатель.

**Для цикла с параметром:** При вызове в функцию ***For\_cond*** передаются значения *ε*, *Y*0, *Y*1, *Y*2 и указатель на параметр *n*. На основе этих значений для начального значения параметра *n*=3 вычисляется значение *Y* по заданной рекуррентной формуле. Проверяется выполнение условия прекращения вычислений ⎪*Y*–*Y*2⎪≤*ε*. Для контроля числа выполненных итераций одновременно проверяется выполнение условия *n*<INT\_MAX.

Если ⎪*Y*–*Y*2⎪>*ε*, для вычисления следующего значения *Y* увеличивается на 1 значение параметра *n*, значение *Y*1 переписывается в *Y*0, значение *Y*2 переписывается в *Y*1, значение *Y* переписывается в *Y*2 и снова вычисляется значение *Y*. Действия повторяются до выполнения условия ⎪*Y*–*Y*2⎪≤*ε*. Необходимо обратить внимание, что очередное значение *Y* всегда вычисляется **непосредственно перед проверкой** выполнения условия прекращения вычислений.

При завершении работы функция возвращает значение *Y*. Значение параметра *n* возвращается через указатель.

***Схема алгоритма решения с использованием цикла while (функция Before\_cond)***

*Before\_cond*(*y*0,*y*1,*y*2,*eps*,&*n*)

y=(5\*sin(y0)+y2)/(3.5\*sqrt(n\*n+1)+2.5\*sqrt(n\*n-1))

|*y*-*y*2|>*eps*&&

*n*<INT\_MAX

нет

да

*n*++

*y*0=*y*1

*y*1=*y*2

*y*2=*y*

y=(5\*sin(y0)+y2)/(3.5\*sqrt(n\*n+1)+2.5\*sqrt(n\*n-1))

*return*(*y*)

***Схема алгоритма решения с использованием цикла do…while (функция After\_cond)***

*After\_cond*(*y*1,*y*2,*y*,*eps*,&*n*)

*n*++

*y*0=*y*1

*y*1=*y*2

*y*2=*y*

y=(5\*sin(y0)+y2)/(3.5\*sqrt(n\*n+1)+2.5\*sqrt(n\*n-1))

|*y*-*y*2|>*eps*&&

*n*<INT\_MAX

да

нет

*return*(*y*)

***Схема алгоритма решения с использованием цикла for (функция For\_cond)***

*For\_cond*(*y*0,*y*1,*y*2,*eps*,&*n*)

y=(5\*sin(y0)+y2)/(3.5\*sqrt(n\*n+1)+2.5\*sqrt(n\*n-1))

; |*y*-*y*2|>*eps*

&&*n*<INT\_MAX;

*n*++

*y*0=*y*1

*y*1=*y*2

*y*2=*y*

y=(5\*sin(y0)+y2)/(3.5\*sqrt(n\*n+1)+2.5\*sqrt(n\*n-1))

*return*(*y*)

**Текст программы**

#include <iostream>

#include <math.h>

using namespace std;

double After\_cond(double eps, double y1, double y2, double y, int& n);

double Before\_cond(double eps, double y0, double y1, double y2, int& n);

double For\_cond(double eps, double y0, double y1, double y2, int& n);

int main()

{

int var,

n1, n2, n3; //число итераций для разных циклов

double re1, re2, re3, //результаты расчетов

Y, //значение предела

Y0, Y1, Y2; //текущие значения элементов последовательности

double EPS; //точность вычисления предела

setlocale(LC\_ALL, "Russian");

for (;;)

{

cout << " Вид действия:" << endl;

cout << " 1 - вычисление предела последовательности" << endl;

cout << " 2 - завершение задачи" << endl;

cout << " Введите вид действия -> ";

cin >> var;

switch (var)

{

case 1:

//ввод данных

cout << "Введите значение точности вычисления предела (EPS), значения элементов последовательности (Y0, Y1, Y2) ";

cin >> EPS >> Y0 >> Y1 >> Y2;

cout << "\n";

//

if (EPS <= 0 || EPS > .1)

{

cout << "Ошибка ввода. Значение eps д.б. >0 и <=0.1\n" << endl;

continue;

}

//Решение циклом while

re1 = Before\_cond(EPS, Y0, Y1, Y2, n1 = 3);

if (n1 != INT\_MAX)

{

cout.precision(4); //число знаков после десятичной точки

cout << "Для цикла WHILE рез. = " << re1 << " n = " << n1 << endl;

}

else cout << "Для цикла WHILE точность не достигнута\n";

//Решение циклом do..while

re2 = After\_cond(EPS, Y0, Y1, Y2, n2 = 2);

if (n2 != INT\_MAX)

{

cout.precision(4); //число знаков после десятичной точки

cout << "Для цикла DO..WHILE рез.= " << re2 << " n = " << n2 << endl;

}

else cout << "Для цикла DO..WHILE точность не достигнута\n";

re3 = For\_cond(EPS, Y0, Y1, Y2, n3 = 3);

if (n3 != INT\_MAX)

{

cout.precision(4); //число знаков после десятичной точки

cout << "Для цикла FOR рез. = " << re3 << " n = " << n3 << endl;

}

else cout << "Для цикла FOR точность не достигнута\n";

break;

default: return 0;

}

}

}

double Before\_cond(double eps, double y0, double y1, double y2, int &n)

{

double y = (5 \* sin(y0) + y2) / (3.5 \* sqrt(n \* n + 1) + 2.5 \* sqrt(n \* n - 1));

while(fabs(y - y2) > eps && n < INT16\_MAX)

{

y0 = y1;

y1 = y2;

y2 = y;

n++;

y = (5 \* sin(y0) + y2) / (3.5 \* sqrt(n \* n + 1) + 2.5 \* sqrt(n \* n - 1));

}

return y;

}

double After\_cond(double eps, double y1, double y2, double y, int& n)

{

double y0;

do

{

y0 = y1;

y1 = y2;

y2 = y;

n++;

y = (5 \* sin(y0) + y2) / (3.5 \* sqrt(n \* n + 1) + 2.5 \* sqrt(n \* n - 1));

} while (fabs(y - y2) > eps && n < INT16\_MAX);

return y;

}

double For\_cond(double eps, double y0, double y1, double y2, int& n)

{

double y = (5 \* sin(y0) + y2) / (3.5 \* sqrt(n \* n + 1) + 2.5 \* sqrt(n \* n - 1));

for (; fabs(y - y2) > eps && n < INT16\_MAX;)

{

y0 = y1;

y1 = y2;

y2 = y;

n++;

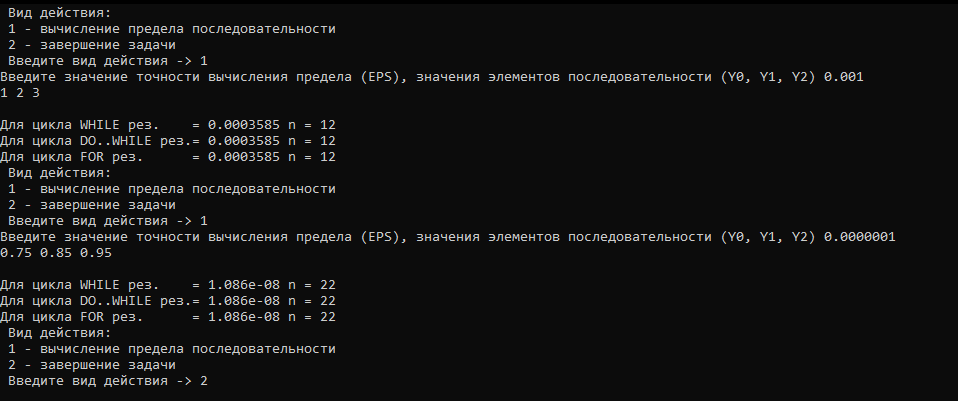
y = (5 \* sin(y0) + y2) / (3.5 \* sqrt(n \* n + 1) + 2.5 \* sqrt(n \* n - 1));

}

return y;

}

**Пример работы программы**

****

**Вывод:** используя технологию процедурного программирования разработал программу решения индивидуальной задачи тремя видами циклических управляющих структур: Цикл - Пока (с предусловием), Цикл - До (с постусловием), Цикл - Для (с параметром).