высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт: Институт цифрового развития

Кафедра: Межинститутская базовая кафедра

**Отчёт по Лабораторной работе № 4 по информационным технологиям и программированию**

выполнил:

Студент, 1 курс, группа ПИЖ-б-о-23-2

направление подготовки

09.03.04 Программная инженерия,

направленность «Разработка и сопровождение программного обеспечения »

форма обучения очная

Черников Дмитрий Дмитриевич

Ставрополь, 2023г

Лабораторная работа № 3. Решение задач с использованием функций

**Цель работы**: научиться программировать задачи с использованием функции

**Вариант – 11(1)**

**Задание 1**. Разработать программы вычисления числовых и функциональных рядов

**Задача 1.1**. Для бесконечного числового ряда члены которого вычисляются по формуле, соответствующей индивидуальному варианту, вычислить сумму первых n членов ряда с использованием функции double Sum(int n)

Изображение выглядит как текст, Шрифт, белый, дизайн

Автоматически созданное описание

Код программы:

#include <iostream>

using namespace std;

int getN() {

int n;

cout << "n = ";

cin >> n;

return n;

}

double sum1() {

int n = getN();

double summa = 0;

for (long long i = 1; i <= n; i++) {

summa += ( i + 1.0) / (i\*i\*i + 2.0 \* i);

}

return summa;

}

int main() {

cout << "Sum1 = " << sum1() << endl;Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

**Задача 1.2.** Для бесконечного числового ряда члены которого вычисляются по формуле, соответствующей индивидуальному варианту, вычислить приближенную сумму двумя способами: сумму первых n членов ряда и сумму с точностью до eps (эпсилон) в двух разных функциях: double Sum1(int n); double Sum2(double eps). Запрещается использование функции pow() и условных операторов «if» и «? : »

Изображение выглядит как текст, Шрифт, белый, типография

Автоматически созданное описание

Код программы:

#include <iostream>

using namespace std;

int getN() {

int n;

cout << "n = ";

cin >> n;

return n;

}

double getEps() {

double eps;

cout << "eps = ";

cin >> eps;

return eps;

}

int znak(int n) {

int res = -1;

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

res \*= -1;

}

return res;

}

double sum1() {

int n = getN();

double summa = 0;

double a\_n = 0;

for (long long i = 1; i <= n; i++) {

a\_n = znak(i) \* (1.0) / (i \* i - i + 2.0);

summa += a\_n;

}

return summa;

}

double sum2 () {

double eps = getEps();

int i = 1;

double sum = 0;

double a\_n = 0;

while (true) {

a\_n = znak(i) \* (1.0) / (i \* i - i + 2.0);

if (abs(a\_n) < eps) {

break;

}

sum += a\_n;

i += 1;

}

return sum;

}

int main() {

cout << "Sum1 = " << sum1() << endl;

cout << "Sum2 = " << sum2();

}**Задание 1.3.** Для функционального ряда коэффициенты которого вычисляются по формуле, соответствующей индивидуальному варианту, вычислить приближенную сумму при фиксированном значении x двумя способами: сумму первых n членов ряда и сумму с точностью до eps (эпсилон) в двух разных функциях: double Sum1(double x, int n); double Sum2(double x, double eps). Запрещается использование условных операторов if и «? :», функции pow(), вторичное вычисление факториала и возведение в степень на каждом шаге итерации (использовать рекуррентные соотношения n!=n·(n-1)!, an =a·an-1 ).

Изображение выглядит как текст, Шрифт, белый, типография

Автоматически созданное описание

Код программы:

#include <iostream>

using namespace std;

int getN() {

int n;

cout << "n = ";

cin >> n;

return n;

}

double getEps() {

double eps;

cout << "eps = ";

cin >> eps;

return eps;

}

int znak(int n) {

int res = -1;

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

res \*= -1;

}

return res;

}

long long MyPow(int num, int n) {

long long result = num;

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

result \*= num;

}

return result;

}

long long fact(int num) {

long long result = 1;

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

result \*= i;

}

return result;

}

double sum1(int n, double x) {

double summa = 0;

for (long long i = 1; i <= n; i++) {

summa += MyPow(x,i) \* znak(i) \* ((i + 1.0) / (MyPow(3,i) \* fact(i)));

}

return summa;

}

double sum2(double eps, int x) {

double summa = 0;

int n = 1;

double a\_n = 0;

while (true) {

a\_n = MyPow(x, n) \* znak(n) \* ((n + 1.0) / (MyPow(3, n) \* fact(n)));

if (abs(a\_n) < eps) {

break;

}

summa += a\_n;

n += 1;

}

return summa;

}

int main() {

double x;

cout << "x = ";

cin >> x;

cout << "Sum1 = " << sum1(getN(), x) << endl;

cout << "Sum2 = " << sum2(getEps(), x);

} }}Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

**Задание 2.** Вычисление характеристик числовых последовательностей бесконечного ряда

**Задание 2.1.** Последовательность данных не всегда нужно сохранять в памяти. Поэтому нужно уметь обрабатывать последовательность по мере поступления ее элементов: при чтении файла, при вводе некоторых данных с клавиатуры и т.д. Пусть имеется последовательность произвольных целых ненулевых чисел, завершающаяся числом 0 (число 0 в последовательность не входит). Количество элементов в последовательности заранее неизвестно, поэтому в данной работе использовать массивы (статические и динамические) нельзя.

11. Найти сумму нечетных чисел последовательности, сумма цифр в которых превышает заданное число P.

Код программы:

#include <iostream>

using namespace std;

int GetP() {

int p;

cout << "p = ";

cin >> p;

return p;

}

bool Check(int num, int p) {

int suma = Sum1(num);

if (suma <= p) {

return true;

}

else {

return false;

}

}

int Sum1(int num) {

int suma = 0;

while (num > 0) {

suma += num % 10;

num /= 10;

}

return suma;

}

int Count1(int p) {

int count = 0;

int now;

while (true) {

cin >> now;

if (now == 0) {

break;

}

if (now % 2 == 1) {

if (Check(now, p)) {

count += 1;

}

}

}

return count;

}

int main() {

cout << Count1(GetP());

}Изображение выглядит как текст, снимок экрана, черный

Автоматически созданное описание

**Задача 2.2**

11.Найти такие элементы (а также их сумму), цифры в которых не превосходят заданного числа М (0<M<9).

Код программы:

#include <iostream>

#include <string>

using namespace std;

int GetM() {

int m;

cout << "M = ";

cin >> m;

return m;

}

bool Check(int num ,int m) {

while (num > 0) {

if (num % 10 > m) {

return false;

}

num /= 10;

}

return true;

}

void Task2\_2(int m) {

int summa = 0;

string nums;

int now;

while (true) {

cin >> now;

if (now == 0) {

break;

}

if (Check(now,m)) {

summa += now;

nums += to\_string(now) + " ";

}

}

cout << "Sum = " << summa << endl << "nums : " << nums;

}

int main() {

Task2\_2(GetM());

}Изображение выглядит как текст, снимок экрана, Шрифт, черный

Автоматически созданное описание

# Задание 2.3.

В последовательности целых чисел найти максимальное количество чисел, идущих подряд, которые обладают свойством Q, и максимальное количество чисел, идущих подряд, которые не обладают свойством Q. Свойство Q задается в варианте. Программа должна содержать логическую функцию, проверяющую, обладает ли заданное число свойством Q.

11 Q: сумма цифр числа является четной

Код программы:

#include <iostream>

using namespace std;

bool Check(int num) {

int sum = 0;

while (num > 0) {

sum += num % 10;

num /= 10;

}

if (sum % 2 == 0) {

return true;

}

return false;

}

void Task2\_3() {

int evenCount = 0;

int evenCountMax = 0;

int oddCount = 0;

int oddCountMax = 0;

int now;

while (true) {

cin >> now;

if (now == 0) {

if (oddCount > oddCountMax) {

oddCountMax = oddCount;

}

if (evenCount > evenCountMax) {

evenCountMax = evenCount;

}

break;

}

if (Check(now)) {

evenCount += 1;

if (oddCount > oddCountMax) {

oddCountMax = oddCount;

}

oddCount = 0;

}

else {

oddCount += 1;

if (evenCount > evenCountMax) {

evenCountMax = evenCount;

}

evenCount = 0;

}

}

cout << "even = " << evenCountMax << endl;

cout << "odd = " << oddCountMax << endl;

}

int main() {

Task2\_3();

}

Изображение выглядит как текст, снимок экрана, Шрифт, черный

Автоматически созданное описание

**Контрольные вопросы**

* 1. **Дайте определение понятию функции в программировании на С++**

*Функция в языке C++* - это по сути группа операторов, которые объединены вместе и выполняют какую-то определённую задачу. Функция всегда имеет имя и опционально может иметь входные параметры. Также для функции всегда задаётся тип возвращаемого значения - это тип данных для того результата вычислений, который функция в буквальном смысле возвращает после выполнения группы входящих в неё операторов.

* 1. **Как определяются функции в С++?**

Определение функции состоит из объявления и текста функции, заключенного в фигурные скобки, которые содержат объявления переменных, операторы и выражения.

**3. В каком порядке определяются функции?**

1. Функция main() — это функция, которая запускается при запуске программы. Она является точкой входа и обычно содержит код для инициализации программы.

2. Другие глобальные функции — это функции, которые не определены внутри какого-либо модуля (например, функции, определенные в стандартной библиотеке). Они могут быть вызваны из любой части программы.

3. Функции, определенные внутри модулей (например, функции, определенные в файлах .cpp) — эти функции могут быть вызваны только из той же модули, где они определены.

4. Локальные функции — это функции, определенные внутри функций main() или других глобальных функций. Они могут быть вызваны только из тех же функций, в которых они определены.

5. Статические функции — это функции, которые объявлены без указания места определения (то есть без указания файла, в котором они определены). Они могут быть вызваны из любой части программы.

6. Динамические функции — это функции, которые объявлены с помощью оператора new. Они могут быть вызваны только из той же области видимости, где они определены.

**4. Опишите процесс вызова и выполнения функции**

1. Вызов функции — это процесс передачи аргументов в функцию и вызова ее имени.

2. Выполнение функции — это процесс выполнения кода, который находится внутри функции. Когда функция вызывается, компилятор выполняет ее код. Код функции выполняется в том порядке, в котором он был написан. Если функция использует какие-либо переменные, они должны быть инициализированы перед выполнением кода.

**5. Можно изменить порядок определения функции?**

Да, в C++ можно изменить порядок определения функций. Однако, важно помнить, что порядок определения функций может влиять на их поведение и использование в программе. Поэтому рекомендуется следовать определенному порядку определения функций, чтобы избежать ошибок и улучшить читаемость кода.

**6. Какие переменные называются локальными и глобальными?**

Локальные переменные — это переменные, которые определены внутри функции и доступны только внутри этой функции. Они имеют область видимости, ограниченную функцией, в которой они определены. Если переменная не была объявлена как глобальная, она будет считаться локальной.

Глобальные переменные — это переменные, которые определены вне функции и доступны из любой части программы. Они имеют область видимости, которая распространяется на всю программу. Если переменная не была объявлена как локальная, она будет считаться глобальной.

**7. Какие переменные при работе с функциями называются формальными и фактическими?**

Формальные переменные — это переменные, которые объявлены в определении функции. Они являются теми переменными, которые используются внутри функции для хранения значений. Формальные переменные имеют область видимости, ограниченную функцией, в которой они объявлены.

Фактические переменные — это переменные, которые используются в теле функции. Они имеют ту же область видимости, что и формальные переменные, и могут быть использованы в любом месте функции. Фактические переменные могут быть определены внутри функции или вне нее.