|  |  |
| --- | --- |
| http://www.pl130.ru/doc/index/%D0%9B%D0%BE%D0%B3%D0%BE%D1%82%D0%B8%D0%BF22.jpg | Санкт-Петербургское государственное бюджетное  профессиональное образовательное учреждение  "Колледж электроники и приборостроения" |

отчет

**по практической работе № 13**

**по дисциплине «Основы алгоритмизации и программирования»**

Тема: Организация функций

|  |  |  |
| --- | --- | --- |
| Студент гр. 13 ИТ |  | Липинский К.С. |
| Преподаватель |  | Счастливцев А.К. |

Санкт-Петербург

2023

**Цель работы.**

Научиться организации функций.

**Выполнение работы.**

**Задание 1.**

using static System.Console;

namespace TestProject

{

class Program

{

static double Average(double a, double b)

{

double avg = (a + b) / 2.0;

return avg;

}

static void Main()

{

double res = Average(17, 44);

WriteLine(res);

}

}

}

****

**Задание 2.**

using static System.Console;

namespace TestProject

{

class Program

{

private static bool CheckCompositivity(int N)

{

for (int i = 2; i <= N / 2; i++)

{

if (N % i == 0)

{

return false;

}

}

return true;

}

static void Main()

{

bool commandStatus = true;

while(commandStatus)

{

try

{

Write("Enter any natural number: ");

int N = Convert.ToInt32(ReadLine());

if (N < 1)

{

throw new ArgumentOutOfRangeException();

}

Write("Enter the range start number: ");

int a = Convert.ToInt32(ReadLine());

if (a < 1)

{

throw new ArgumentOutOfRangeException();

}

Write("Enter the range end number: ");

int b = Convert.ToInt32(ReadLine());

if (b < 1)

{

throw new ArgumentOutOfRangeException();

}

commandStatus = false;

WriteLine($"\nIs number simple: {CheckCompositivity(N)}");

Write("All the simple numbers: ");

for (int i = a; i <= b; i++)

{

if (CheckCompositivity(i))

{

Write($"{i} ");

}

}

int simpCount = 0;

for (int i = a; i <= b; i++)

{

if (!CheckCompositivity(i))

{

simpCount += i;

}

}

WriteLine($"The sum of all composite numbers: {simpCount}");

}

catch (ArgumentOutOfRangeException)

{

BackgroundColor = ConsoleColor.Red;

WriteLine("The number is not natural. Try again\n");

BackgroundColor = default;

}

catch (FormatException)

{

BackgroundColor = ConsoleColor.Red;

WriteLine("Not a correct format. Try again\n");

BackgroundColor = default;

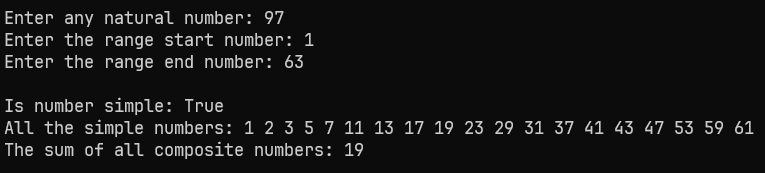
}

}

}

}

}

****

**Задание 3.**

using static System.Console;

namespace TestProject

{

class Program

{

private static int SumDig(int N)

{

int sum = 0;

while (N > 0)

{

sum += N % 10;

N /= 10;

}

return sum;

}

static void Main()

{

Write("Enter any natural number: ");

int N = Convert.ToInt32(ReadLine());

Write("Enter natural constant number: ");

int C = Convert.ToInt32(ReadLine());

Write("Enter range start: ");

int a = Convert.ToInt32(ReadLine());

Write("Enter range end: ");

int b = Convert.ToInt32(ReadLine());

WriteLine($"\nSum of digits: {SumDig(N)}");

Write("Numbers with digits' sum equal to C: ");

for (int i = a; i <= b; i++)

{

if (SumDig(i) == C)

{

Write($"{i} ");

}

}

Write("\nNumbers with uneven digits' sum: ");

for (int i = a; i <= b; i++)

{

if (SumDig(i) % 2 != 0)

{

Write($"{i} ");

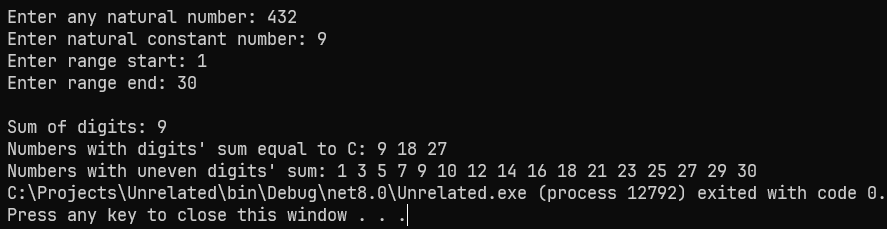
}

}

}

}

}



**Задание 4.**

using static System.Console;

namespace Prac

{

class Program

{

private static double Max(double x, double y)

{

return x > y ? x : y;

}

private static double SegmentLength(double x1, double y1, double x2, double y2)

{

return Math.Sqrt(Math.Pow((x2 - x1), 2) + Math.Pow((y2 - y1), 2));

}

private static string Farthest(double x1, double y1, double x2, double y2, double x3, double y3)

{

return "None";

}

static void Main()

{

Write("Enter x1: "); double x1 = Convert.ToDouble(ReadLine());

Write("Enter y1: "); double y1 = Convert.ToDouble(ReadLine());

Write("Enter x2: "); double x2 = Convert.ToDouble(ReadLine());

Write("Enter y2: "); double y2 = Convert.ToDouble(ReadLine());

Write("Enter x3: "); double x3 = Convert.ToDouble(ReadLine());

Write("Enter y3: "); double y3 = Convert.ToDouble(ReadLine());

WriteLine($"\nLength of the segment: {SegmentLength(x1, y1, x2, y2):F2}");

double dist1 = SegmentLength(0, 0, x1, y1);

double dist2 = SegmentLength(0, 0, x2, y2);

double dist3 = SegmentLength(0, 0, x3, y3);

double farthest = Max(dist1, Max(dist2, dist3));

Write("Farthest point: ");

if (farthest == dist1)

{

Write("point 1");

}

else if (farthest == dist2)

{

Write("point 2");

}

else if (farthest == dist3)

{

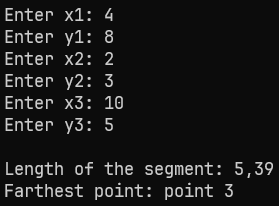
Write("point 3");

}

}

}

}



**Вопросы.**

1. Тело метода в C# — это блок кода, который содержит определение и реализацию операций, которые должны быть выполнены при вызове метода.

2. Особенности метода-процедуры:

- Можно не возвращать значение, если возвращаемый тип – void.

- Может иметь входные параметры.

3. В теле метода может отсутствовать оператор return, если тип возвращаемого значения void (т.е. ничего не возвращается).