**Лабораторная работа №1. Вспомогательные функции**

**Цель работы:** приобретение навыков составления и отладки программ с использованием пользовательских функций для замера продолжительности процесса вычисления.

**Ход работы**

## **Написание кода программы.**

Для проверки работоспособности разработанных функций и приобретения навыков замера продолжительности процесса вычисления была реализована программа приведенная в листинге Lab1.cpp.

Код файлов Lab1.cpp приведен в листинге 1.1.

|  |
| --- |
| #include "Auxil.h" // вспомогательные функции  #include <iostream>  #include "Fibanachi.h"  #include <ctime>  #include <locale>  #define CYCLE 1000000 // количество циклов  #define I 10  #define R 40  int main()  {  setlocale(LC\_ALL, "rus");  double av1 = 0, av2 = 0;  clock\_t t1 = 0, t2 = 0;  auxil::start(); // старт генерации  t1 = clock(); // фиксация времени  for (int i = 0; i < CYCLE; i++)  {  av1 += (double)auxil::iget(-100, 100); // сумма случайных чисел  av2 += auxil::dget(-100, 100); // сумма случайных чисел  }  t2 = clock(); // фиксация времени  std::cout << std::endl << "количество циклов: " << CYCLE;  std::cout << std::endl << "среднее значение (int): " << av1 / CYCLE;  std::cout << std::endl << "среднее значение (double): " << av2 / CYCLE;  std::cout << std::endl << "продолжительность (у.е): " << (t2 - t1);  std::cout << std::endl << " (сек): "  << ((double)(t2 - t1)) / ((double)CLOCKS\_PER\_SEC);  std::cout << std::endl;  std::cout << std::endl << "Сумма случайных чисел:";  av1 = 0; av2 = 0;  clock\_t SS1[I];  int iter;  auxil::start();  for (int n = 0; n < I; n++)  {  iter = CYCLE + n \* CYCLE;  SS1[n] = clock();  for (int i = 0; i < iter; i++)  {  av1 += (double)auxil::iget(-100, 100); // сумма случайных чисел  av2 += auxil::dget(-100, 100); // сумма случайных чисел  }    SS1[n] = clock() - SS1[n];  std::cout << std::endl << "n = " << n + 1 << ": " << SS1[n];  }  std::cout << std::endl;  std::cout << std::endl << "Числа Фибаначи: ";  clock\_t SS2[R];  for (int n = 0; n < R; n++)  {  SS2[n] = clock();  Fibanachi(n + 1);  SS2[n] = clock() - SS2[n];  std::cout << std::endl << "n = " << n + 1 << ": " << SS2[n];  }  std::cout << std::endl;  system("pause");  return 0;  } |

Листинг 1.1. Содержание файла Lab1.cpp

***Задание 1.*** Разработайте три функции (start, dget и iget) в Auxil.h

разработанные функции должны располагаться в файле **Auxil.cpp**, а в файле **Auxil.h –** прототипы функций

Код файла Auxil.h приведен в листинге 1.3.

|  |
| --- |
| #pragma once  #include <cstdlib>  namespace auxil  {  void start(); // старт генератора сл. чисел  double dget(double rmin, double rmax); // получить случайное число  int iget(int rmin, int rmax); // получить случайное число  }; |

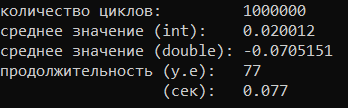
Листинг 1.3. Содержание файла Auxil.h

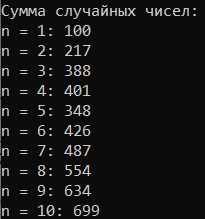
Код файла Auxil.cpp приведет в листинге 1.4.

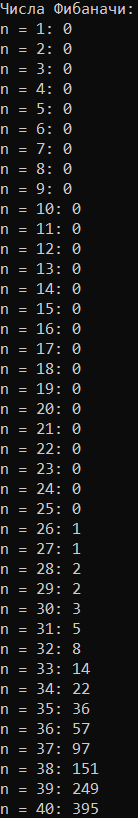
|  |
| --- |
| #include "stdafx.h"  #include "Auxil.h"  #include <ctime>  namespace auxil  {  void start() // старт генератора сл. чисел  {  srand((unsigned)time(NULL));  };  double dget(double rmin, double rmax) // получить случайное число  {  return ((double)rand() / (double)RAND\_MAX) \* (rmax - rmin) + rmin;  };  int iget(int rmin, int rmax) // получить случайное число  {  return (int)dget((double)rmin, (double)rmax);  };  } |

Листинг 1.4. Содержание файла Auxil.cpp

Результат работы программы представлен на рисунке 1.1.







## **1.2. Измерение скорости выполнения функции генерации случайных чисел.**

Результаты измерений и соответствующий график приведены на рисунке 1.2.



Рисунок 1.2. Результаты измерений и их график

Вывод: скорость выполнения программы линейно зависит от количества итераций цикла.

## **1.3. Измерение скорости выполнения функции для решения чисел Фибоначи.**

Результаты измерений и соответствующий график приведены на рисунке 1.3.



Рисунок 1.3. Результаты измерений и их график

Вывод: скорость выполнения программы экспоненциально зависит от числа Фибоначчи.