Федеральное государственное образовательное бюджетное учреждение

высшего профессионального образования

«Нижегородский Государственный Университет им.

Н.И.Лобачевского» (ННГУ)

Институт Информационных Технологий Математики и Механики

Отчёт по лабораторной работы

Работа с массивами случайно сгенерированных чисел

Выполнил:

студент группы 3821Б1ФИ3

Вьюнов Д.В.

Проверил:

заведующий лабораторией суперкомпьютерных технологий и высокопроизводительных вычислений

Лебедев И.Г

Нижний Новгород

2021г.

Содержание

[Введение 2](#_Toc85632832)

[Постановка задачи 3](#_Toc85632833)

[Руководство пользователя 4](#_Toc85632834)

[Руководство программиста 5](#_Toc85632835)

[Описание структуры кода программы 5](#_Toc85632836)

[Описание структуры данных 7](#_Toc85632837)

[Описание алгоритмов 7](#_Toc85632838)

[Эксперименты 10](#_Toc85632839)

[Заключение 13](#_Toc85632840)

[Список литературы 14](#_Toc85632841)

[Приложение 1 15](#_Toc85632842)

# Введение

Программирование — это интересный, полезный и увлекательный процесс, благодаря которому создаются программы – набор инструкций, которые приводятся в исполнение компьютерами.

Случайные числа — это одна из основных составляющих любого языка программирования, на них строятся многие алгоритмы. Они имеют применение в физике, например, в исследованиях электронного шума, в инженерном деле и исследовании операций. Многие методы статистического анализа требуют использования случайных чисел.

В ходе выполнения лабораторный работы на языке программирования «С» будет написана программа, работающая со случайными числами.

# Постановка задачи

Программа генерирует множество случайных чисел размера n в диапазоне (min, max), где n, min, max вводятся с клавиатуры. После чего подсчитывает, выводит сумму, которая получается следующим образом: все числа, номера которых совпадают с дробной частью одного из исходных чисел - вычитаются, все остальные прибавляются.

# Руководство пользователя

После запуска программа выводит сообщения “Input count of numbers”, “Count of number after decimal point”, “Minimum of random number”, “Maximum of random number”, после которых нужно ввести соответствующие им значения.

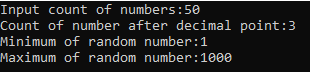


Рисунок 1. Ввод данных от пользователя

После нажатии Enter, программа выведет результат суммирования сгенерированных чисел, произведенного по правилам технического задания. На этом программа завершается (см. рис. 3).

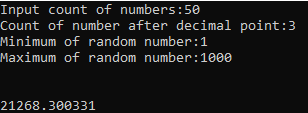


Рисунок 2. Результат работы программы

# Руководство программиста

## Описание структуры кода программы

1. Подключение библиотек, с которыми предстоит работать.

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <time.h>  #include <math.h> |

1. Объявление функции get\_rand, возвращающая рандомное число от 0 до 1.

|  |
| --- |
| double get\_rand()  {  return rand() / RAND\_MAX\_F;  } |

1. Объявление функции get\_rand\_range, возвращающая рандомное число от min до max.

|  |
| --- |
| double get\_rand\_range(const double min, const double max)  {  return get\_rand() \* (max - min) + min;  } |

1. Объявление функции main, вначале которой мы объявляем переменные. Используем srand() для генерации случайного числа.

|  |
| --- |
| int main()  {  srand(time(NULL));  float minimum, maximum;  double\* arr, sum = 0;  int ost, arr\_size, degree; |

1. Получение данных от пользователя и проверка их на корректность.

|  |
| --- |
| printf("Input count of numbers:");  scanf\_s("%d", &arr\_size);  printf("Count of number after decimal point:");  scanf\_s("%d", &degree);  printf("Minimum of random number:");  scanf\_s("%f", &minimum);  printf("Maximum of random number:");  scanf\_s("%f", &maximum);  printf("\n");  arr = (double\*)malloc(arr\_size \* sizeof(double));  if (minimum > maximum || arr\_size <= 0)  {  printf("Input is wrong");  return 1;  } |

1. Заполнение массива arr\_size рандомными числами в диапазоне от min до max.

|  |
| --- |
| for (int i = 0; i < arr\_size; i++) // Заполняем массив  {  arr[i] = get\_rand\_range(minimum, maximum);  } |

## Отделяем дробную часть числа, посредством вычитания из исходного числа его целой копии и умножением этой разности на 10 в степени degree, для отделения degree знаков после запятой. Проверяем, есть ли индекс в массиве, соответствующий дробной части числа. Если такой есть, то зануляем число под этим индексом. Следующим циклом суммируем все числа, индексы которых не совпадают с дробной частью ни одного из чисел

|  |
| --- |
| for (int j = 0; j < arr\_size; j++)  {  if (arr[j] < 0) {  ost = (arr[j] + (int)arr[j] \* (-1)) \* pow(10, degree) \* (-1);  } else  ost = (arr[j] - (int)arr[j]) \* pow(10, degree);  if (ost < arr\_size)  {  arr[ost] = 0;  }  }  for (int k = 0; k < arr\_size; k++)  sum += arr[k]; |

1. Печатаем значение переменной sum.

|  |
| --- |
| printf("\n%lf", sum);  return 0;  } |

## Описание структуры данных

* В программе используются переменные счетчики типа int(i, j), количество генерируемых чисел arr\_size типа int, а также функция get\_rand\_range(), возвращающая число типа double.
* Переменные min, max относятся к типу данных float.
* Случайные числа, создаваемые в ходе работы программы, записываются в массив типа double arr, а дробные части этих чисел записываютcя в переменную типа int ost.
* А также для функционирования программы требуются библиотеки: stdio.h, stdlib.h, time.h.

## Описание алгоритмов

В данной программе реализовано несколько ключевых алгоритмов:

* Выделение дробной части с фиксированной точностью и представление её как целого числа. Для из исходного числа вычитается его целая часть, полученное число умножается на 10 в степени degree (количество знаков после запятой), приведение нового числа к типа int и присвоение переменной ost значение полученного числа.

|  |
| --- |
| for (int j = 0; j < arr\_size; j++)  {  if (arr[j] < 0)  {  ost = (arr[j] + (int)arr[j] \* (-1)) \* pow(10, degree) \* (-1);  } else  ost = (arr[j] - (int)arr[j]) \* pow(10, degree); |

* Алгоритм заполнения массива случайными числами представлен следующей блок-схемой:

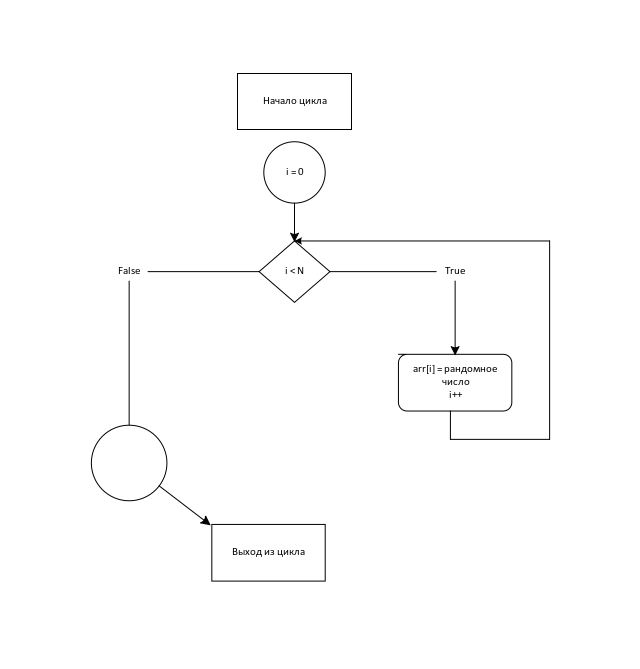


Рисунок 3. Блок-схема заполнения массива случайными числами

* Алгоритм суммирования.

|  |
| --- |
| if (ost < arr\_size)  {  arr[ost] = 0;  }  }  for (int k = 0; k < arr\_size; k++)  sum += arr[k];  printf("%lf", sum); |

# Эксперименты

Написанную программу протестируем на различных входных данных, чтобы убедится в её работоспособности в различных условиях.

1. В случае ввода некорректных данных, таких как отрицательное количество элементов или неверный ввод границ диапазона (см. рис. 4).

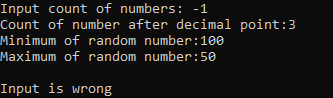


Рисунок 4. Ввод неверных данных

1. Работа программы в случае ввода корректных данных (см. рис. 5).

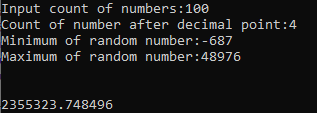


Рисунок 5. Результат работы программы при корректных входных данных

Теперь попробуем подать на вход различные корректные значения (см. рис. 6-7).

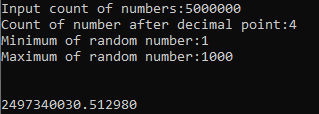


Рисунок 6. Работа программы при больших N

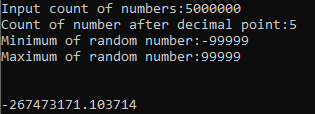


Рисунок 7. Работа программы при больших входных данных

При тестировании программы с различными входными данными в среде разработки Visual Studio никаких ошибок или предупреждений встречено не было.

# Заключение

В ходе лабораторной работы мною была написана программа на языке «С», которая генерирует множество случайных чисел размера n в диапазоне (min, max), где n, min, max вводятся с клавиатуры. После чего подсчитывает, выводит сумму, которая получается следующим образом: все числа, номера которых совпадают с дробной частью одного из исходных чисел - вычитаются, все остальные прибавляются.

Выполнение данной лабораторной работы позволило мне лучше изучить принципы работы с динамическими массивами, изучить различные алгоритмы позволяющие работать с данными, содержащимися в массивах, принципы работы со случайными числами, принципы работы с вводимыми пользователем данными их реализации внутри программы, их интерпретацию и проверку на корректность.

Список литературы

1. C/C++. Программирование на языке высокого уровня / Т. А. Павловская. — СПб.: Питер, 2003. —461 с: ил.
2. Сборник задач по программированию/Д.М. Златопольский — 3-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2011. — 304 с.

# Приложение 1

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <time.h>  #include <math.h>  const float RAND\_MAX\_F = RAND\_MAX;  double get\_rand() // Генератор рандомного числа  {  return rand() / RAND\_MAX\_F;  }  double get\_rand\_range(const double min, const double max) // Генератор рандомного числа в диапазоне  {  return get\_rand() \* (max - min) + min;  }  int main()  {  srand(time(NULL)); // Семя для полностью рандомного числа  float minimum, maximum;  double\* arr, sum = 0;  int ost, arr\_size, degree;  printf("Input count of numbers:");  scanf\_s("%d", &arr\_size);  printf("Count of number after decimal point:");  scanf\_s("%d", &degree);  printf("Minimum of random number:");  scanf\_s("%f", &minimum); // Рандомное число от  printf("Maximum of random number:");  scanf\_s("%f", &maximum); // Рандомное число до  printf("\n");  arr = (double\*)malloc(arr\_size \* sizeof(double));  if (minimum > maximum || arr\_size <= 0) // Проверка входных  {  printf("Input is wrong");  return 1;  }  for (int i = 0; i < arr\_size; i++) // Заполняем массив  {  arr[i] = get\_rand\_range(minimum, maximum);  }  for (int j = 0; j < arr\_size; j++) // Отделяем дробную часть и проверяем  {  if (arr[j] < 0) {  ost = (arr[j] + (int)arr[j] \* (-1)) \* pow(10, degree) \* (-1); // Умножаем на 10^(degree) для того чтобы сдвинуть дробную часть влево  }  else  ost = (arr[j] - (int)arr[j]) \* pow(10, degree);  if (ost < arr\_size)  {  arr[ost] = 0;  }  }  for (int k = 0; k < arr\_size; k++)  sum += arr[k];  printf("\n%lf", sum);  return 0;  } |