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 работе

Реализация разных типов сортировок в языке C, работа с файлами.

Выполнил:

студент группы 3821Б1ПМ3

Шубин М. А.

Проверил:

заведующий лабораторией

суперкомпьютерных технологий и

высокопроизводительных вычислений

Лебедев И.Г

Нижний Новгород

2021 г.

Содержание.

[I. Введение. 2](#_Toc9182)

[II. Постановка задачи. 3](#_Toc28753)

[III. Руководство пользователя. 4](#_Toc15731)

[Первая программа. 4](#_Toc7242)

[Вторая программа. 4](#_Toc21503)

[IV. Руководство программиста. 6](#_Toc31589)

[Описание структуры кода программ. 6](#_Toc30739)

[Первая программа. 6](#_Toc19990)

[Вторая программа. 7](#_Toc14811)

[Описание структур данных. 12](#_Toc30549)

[Описание алгоритмов. 12](#_Toc7673)

[V. Эксперименты. 15](#_Toc10372)

[VI. Заключение. 16](#_Toc31343)

[VII. Список литературы. 17](#_Toc10551)

[VIII. Приложение 1. 18](#_Toc945)

[IX. Приложение 2. 19](#_Toc641)

# Введение.

Один из самых важных элементов программирования - реализации алгоритмов в ЭВМ посредством специализированных языков программирования - работа с файлами. Благодаря работе с файлами могут быть составлены алгоритмы, при работе которых участие человека сведено к нулю, однако вместе с тем с помощью записи в файлы можно просмотреть, что получается в результате работы разных этапов алгоритма.

Алгоритмы сортировки в ЭВМ, как и любые другие алгоритмы, работают строго по определённым правилам, имеют сложность и время выполнения. Целью данной работы является анализ эффективности работы трёх алгоритмов сортировки: «пузырьком», вставками и быстрой.

# Постановка задачи.

Сравнение сортировок.

Реализовать сортировки массивов данных (тип данных определяется преподавателем (В данной работе - float)) задаваемых: обязательно случайно, дополнительно с клавиатуры или из файла.

Реализовать сортировки: пузырьком, вставкой, быстрая.

Сравнить время работы, сделать выводы.

Первая программа создаёт текстовый файл с записанными в него числами. Программа принимает количество чисел n, максимальное и минимальное значение.

Вторая программа читает текстовый файл с набором чисел, выводит консольный интерфейс (печать, сортировка, сброс, выход), выполняет выбранные действия.

# Руководство пользователя.

## Первая программа.

При запуске первой программы выведется «n=», далее необходимо ввести значение n - количества генерируемых чисел (оно должно быть больше 0). После ввода значения n программа выведет «min=», далее требуется ввести нижнюю границу диапазона, в котором будут генерироваться числа. Следующим будет выведено «max=», после которого нужно ввести верхнюю границу диапазона. Верхняя граница должна быть не меньше нижней. Вывод терминала сразу после запуска можно найти в рис. 1.



Рисунок 1. Терминал после запуска

Пример ввода: n=5, min=-10 и max=10. (см. рис. 2)

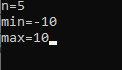


Рисунок 2. Ввод данных от пользователя

После ввода значения max и нажатия клавиши Enter программа выводит сгенерированные числа в файл «output\_1.txt». После нажатия любой клавиши программа завершается (см. рис. 3)



Рисунок 3. Результат работы программы

## Вторая программа.

При запуске второй программы будут выведены возможные команды: 1 - Вывод чисел на экран; 2 - Запуск сортировок; 3 - Выход из программы (см. рис. 4). Программа считывает значения из файла «output\_1.txt», поэтому при его отсутствии в папке с программой будет выведена ошибка (см. рис. 5).

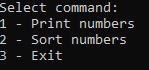


Рисунок 4. Терминал после запуска



Рисунок 5. Ошибка, возникающая при отсутствии файла «output\_1.txt»

Ввод команды 1 выведет числа из файла на экран (см. рис. 6); при вводе команды 2 последовательно будут произведены сортировки тремя вариантами: пузырьком», вставками и быстрой, после каждой будут выведено время выполнения конкретной сортировки, по которому можно оценить, какая сортировка работает быстрее (см. рис. 7); ввод команды 3 завершит программу. Ввод целых чисел, отличных от данных, не приведёт ни к чему. Ввод значений, отличных от целых чисел, приведёт к ошибке программы, поэтому ввод подобных значений запрещён.

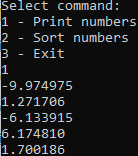


Рисунок 6. Вывод в консоль при вводе команды 1

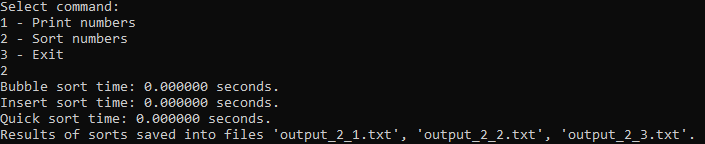


Рисунок 7. Вывод в консоль при вводе команды 2

# Руководство программиста.

## Описание структуры кода программ.

## Первая программа.

1. Подключение библиотек, необходимых для работы.

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <conio.h> |

1. Объявление функции main(), всех используемых переменных, получение данных от пользователя и проверка их на корректность.

|  |
| --- |
| int main()  {    long int n = 0, i = 0;    float mx = 0.0, mn = 0.0, randn = 0.0;    FILE \*file = NULL;    file = fopen("output\_1.txt", "w");    if (file == NULL)    {      printf("Error while opening file 'output\_1.txt'. Press any key and Enter to exit.");      scanf("%d", &n);      return 0;    }    printf("n=");    scanf("%d", &n);    if (n <= 0)    {      printf("Error: n must be greater than 0. Press any key and Enter to exit.");      scanf("%d", &n);      return 0;    }    printf("min=");    scanf("%f", &mn);    printf("max=");    scanf("%f", &mx);  if (mn > mx)    {      printf("Error: min > max. Press any key and Enter to exit.");      scanf("%d", &n);      return 0;    } |

1. Ввод в файл количества чисел, после него - сами числа в каждой строчке, закрытие файла и выход из программы после нажатия пользователем любой клавиши.

|  |
| --- |
| fprintf(file, "%d\n", n);    for(i = 0; i < n; i++)    {      randn = (float)rand() / RAND\_MAX \* (mx - mn) + mn;      fprintf(file, "%f\n", randn);    }    fclose(file);    printf("Generation finished successfully. Press any key to exit.");    getch();  } |

## Вторая программа.

1. Подключение библиотек, необходимых для работы.

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

1. Объявление функций сортировок, введение алгоритмов сортировок (BubbleSort - «пузырьком», InsertSort - вставками, QuickSort - быстрая).

|  |
| --- |
| void BubbleSort(float \*mas, int size)  {    long int i = 0, j = 0;    char noSwap = 1;    float temp = 0.0;    for (i = size - 1; i >= 0; i--)    {      noSwap = 1;      for (j = 0; j < i; j++)      {        if (mas[j] > mas[j + 1])        {          temp = mas[j];          mas[j] = mas[j + 1];          mas[j + 1] = temp;          noSwap = 0;        }      }      if (noSwap == 1)        break;    }  }  void InsertSort(float \*mas, int size)  {    long int i = 0, loc = 0;    float temp = 0.0;    for (i = 1; i < size; i++)    {      temp = mas[i];      loc = i - 1;      while(loc >= 0 && mas[loc] > temp)      {        mas[loc + 1] = mas[loc];        loc--;      }      mas[loc + 1] = temp;    }  }  void QuickSort(float \*mas, int size)  {    int i = 0, j = size - 1;    float mid = mas[(int)(size / 2)];    float temp = 0.0;    do {      while(mas[i] < mid)      {        i++;      }      while(mas[j] > mid)      {        j--;      }      if (i <= j)      {        temp = mas[i];        mas[i] = mas[j];        mas[j] = temp;        i++;        j--;      }    } while (i <= j);    if(j > 0) {      QuickSort(mas, j + 1);    }    if (i < size) {      QuickSort(&mas[i], size - i);    }  } |

1. Объявление функции main(), всех используемых переменных, попытка открыть файл (если файла «output\_1.txt» не существует, пишет в консоль ошибку), считывание значений из файла в массивы mas и его копию masC и закрытие файла.

|  |
| --- |
| int main()  {    long int n = 0, i = 0, j = 0, loc = 0, comm = 0;    float mx = 0.0, mn = 0.0, temp = 0.0;    float readn = 0.0;    float\* mas;    float\* masC;    FILE \*file = NULL;    clock\_t t\_start;    clock\_t t\_end;    file = fopen("output\_1.txt", "r");    if (file == NULL)    {      printf("No file named 'output\_1.txt' found. Press any key to exit.");      getch();      return 0;    }    fscanf(file, "%d", &n);    mas = (float\*)malloc(n \* sizeof(float));    masC = (float\*)malloc(n \* sizeof(float));    for (i = 0; i < n; i++)    {      fscanf(file, "%f", &readn);      mas[i] = readn;      masC[i] = readn;    }    fclose(file); |

1. Считывание команды с консоли до того момента, пока пользователь не введёт 3 (выход). Команды: 1 - Вывод чисел в консоль, 2 - Выполнение сортировок, вывод времени каждой сортировки на экран и создание//перезапись файлов, соответствующих сортировкам («output\_2\_1.txt» - для сортировки «пузырьком», «output\_2\_2.txt» - для сортировки вставками, «output\_2\_3.txt» - для быстрой сортировки), 3 - Выход. В случае ввода любого другого числа не произойдёт ничего. После ввода 3 производится выход из цикла, освобождение массивов и выход из программы.

|  |
| --- |
| while (comm != 3)    {      printf("Select command:\n1 - Print numbers\n2 - Sort numbers\n3 - Exit\n");      scanf("%d", &comm);      switch(comm)      {        case 1:          {            for (i = 0; i < n; i++)            {              printf("%lf\n", masC[i]);            }            printf("\n");            break;          }        case 2:          {            t\_start = clock();            BubbleSort(masC, n);            t\_end = clock();            file = fopen("output\_2\_1.txt", "w");            for (i = 0; i < n; i++)            {              fprintf(file, "%lf\n", masC[i]);              masC[i] = mas[i];            }            fclose(file);            printf("Bubble sort time: %lf seconds.\n", (float)((t\_start - t\_end) / CLOCKS\_PER\_SEC));            t\_start = clock();            InsertSort(masC, n);            t\_end = clock();            file = fopen("output\_2\_2.txt", "w");            for (i = 0; i < n; i++)            {              fprintf(file, "%lf\n", masC[i]);              masC[i] = mas[i];            }            fclose(file);            printf("Insert sort time: %lf seconds.\n", (float)((t\_start - t\_end) / CLOCKS\_PER\_SEC));            t\_start = clock();            QuickSort(masC, n);            t\_end = clock();            file = fopen("output\_2\_3.txt", "w");            for (i = 0; i < n; i++)            {              fprintf(file, "%lf\n", masC[i]);              masC[i] = mas[i];            }            fclose(file);            printf("Quick sort time: %lf seconds.\n", (float)((t\_start - t\_end) / CLOCKS\_PER\_SEC));            printf("Results of sorts saved into files 'output\_2\_1.txt', 'output\_2\_2.txt', 'output\_2\_3.txt'.\n\n");            break;          }        default:          break;      }    }    free(mas);    free(masC);    return 0;  } |

## Описание структур данных.

* В обеих программах использованы стандартные структуры данных, такие как float, long int, char, float\*, FILE.
* Во второй программе использована структура clock\_t из модуля time.h. Данная структура позволяет выполнить подсчёт времени работы сортировок.

## Описание алгоритмов.

В данной программе реализовано три алгоритма сортировки:

* Сортировка «пузырьком»:

|  |
| --- |
| void BubbleSort(float \*mas, int size)  {    long int i = 0, j = 0;    char noSwap = 1;    float temp = 0.0;    for (i = size - 1; i >= 0; i--)    {      noSwap = 1;      for (j = 0; j < i; j++)      {        if (mas[j] > mas[j + 1])        {          temp = mas[j];          mas[j] = mas[j + 1];          mas[j + 1] = temp;          noSwap = 0;        }      }      if (noSwap == 1)        break;    }  } |

* Сортировка вставками:

|  |
| --- |
| void InsertSort(float \*mas, int size)  {    long int i = 0, loc = 0;    float temp = 0.0;    for (i = 1; i < size; i++)    {      temp = mas[i];      loc = i - 1;      while(loc >= 0 && mas[loc] > temp)      {        mas[loc + 1] = mas[loc];        loc--;      }      mas[loc + 1] = temp;    }  } |

* Быстрая сортировка:

|  |
| --- |
| void QuickSort(float \*mas, int size)  {    int i = 0, j = size - 1;    float mid = mas[(int)(size / 2)];    float temp = 0.0;    do {      while(mas[i] < mid)      {        i++;      }      while(mas[j] > mid)      {        j--;      }      if (i <= j)      {        temp = mas[i];        mas[i] = mas[j];        mas[j] = temp;        i++;        j--;      }    } while (i <= j);    if(j > 0) {      QuickSort(mas, j + 1);    }    if (i < size) {      QuickSort(&mas[i], size - i);    }  } |

# Эксперименты.

Необходимо удостовериться в том, что программа корректно работает при любых вариантах ввода.

1. Ввод нулевого (или отрицательного) значения n в первой программе. (см. рис. 8)



Рисунок 8. Ввод неверного значения N

1. Ввод неверных границ min и max в первой программе. (см. рис. 9)

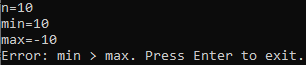


Рисунок 9. Ввод некорректных границ диапазона

1. Вывод первой программы при верно введённых исходных значениях. (см. рис. 10)



Рисунок 10. Работа при верном вводе

1. Файл «output\_1.txt» не был обнаружен второй программой. (см. рис. 11)



Рисунок 11. Входной файл не был обнаружен

Таблица для сравнения времени сортировок.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 10000 эл. | 50000 эл. | 100000 эл. | 500000 эл. |
| Сортировка «пузырьком» | 0 сек. | 6 сек. | 27 сек. | 684 сек. |
| Сортировка вставками | 0 сек. | 1 сек. | 5 сек. | 144 сек. |
| Быстрая сортировка | 0 сек. | 0 сек. | 0 сек. | 0 сек. |

# Заключение.

В ходе лабораторной работы было написано две программы на языке С: одна генерирует n случайных чисел в диапазоне от min до max, где n, min и max вводятся с клавиатуры, и вводит их в файл «output\_1.txt», вторая считывает значения из этого файла и может вывести их на экран, последовательно применить к ним сортировки «пузырьком», вставками и быструю и выводит время выполнения сортировок для сравнения.

Тесты, проведённые с помощью второй программы, позволили выяснить, что за самое короткое время из рассматриваемых сортировок выполняется быстрая, а самая медленная из них - сортировка «пузырьком».

В ходе выполнения данной работы я улучшил свои навыки работы с файлами, функциями и условным оператором switch-case в языке C.

Выполнение подобных работ позволяет проверить свои знания в области выбранного языка, отточить умение написания программ или даже найти более эффективные методы реализация различных функций.

# 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 <conio.h>

int main()

{

  long int n = 0, i = 0;

  float mx = 0.0, mn = 0.0, randn = 0.0;

  FILE \*file = NULL;

  file = fopen("output\_1.txt", "w");

  if (file == NULL)

  {

    printf("Error while opening file 'output\_1.txt'. Press any key and Enter to exit.");

    scanf("%d", &n);

    return 0;

  }

  printf("n=");

  scanf("%d", &n);

  if (n <= 0)

  {

    printf("Error: n must be greater than 0. Press any key and Enter to exit.");

    scanf("%d", &n);

    return 0;

  }

  printf("min=");

  scanf("%f", &mn);

  printf("max=");

  scanf("%f", &mx);

  if (mn > mx)

  {

    printf("Error: min > max. Press any key and Enter to exit.");

    scanf("%d", &n);

    return 0;

  }

  fprintf(file, "%d\n", n);

  for(i = 0; i < n; i++)

  {

    randn = (float)rand() / RAND\_MAX \* (mx - mn) + mn;

    fprintf(file, "%f\n", randn);

  }

  fclose(file);

  printf("Generation finished successfully. Press any key to exit.");

  getch();

}

# Приложение 2.

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

#include <time.h>

void BubbleSort(float \*mas, int size)

{

  long int i = 0, j = 0;

  char noSwap = 1;

  float temp = 0.0;

  for (i = size - 1; i >= 0; i--)

  {

    noSwap = 1;

    for (j = 0; j < i; j++)

    {

      if (mas[j] > mas[j + 1])

      {

        temp = mas[j];

        mas[j] = mas[j + 1];

        mas[j + 1] = temp;

        noSwap = 0;

      }

    }

    if (noSwap == 1)

      break;

  }

}

void InsertSort(float \*mas, int size)

{

  long int i = 0, loc = 0;

  float temp = 0.0;

  for (i = 1; i < size; i++)

  {

    temp = mas[i];

    loc = i - 1;

    while(loc >= 0 && mas[loc] > temp)

    {

      mas[loc + 1] = mas[loc];

      loc--;

    }

    mas[loc + 1] = temp;

  }

}

void QuickSort(float \*mas, int size)

{

  int i = 0, j = size - 1;

  float mid = mas[(int)(size / 2)];

  float temp = 0.0;

  do {

    while(mas[i] < mid)

    {

      i++;

    }

    while(mas[j] > mid)

    {

      j--;

    }

    if (i <= j)

    {

      temp = mas[i];

      mas[i] = mas[j];

      mas[j] = temp;

      i++;

      j--;

    }

  } while (i <= j);

  if(j > 0) {

    QuickSort(mas, j + 1);

  }

  if (i < size) {

    QuickSort(&mas[i], size - i);

  }

}

int main()

{

  long int n = 0, i = 0, j = 0, loc = 0, comm = 0;

  float mx = 0.0, mn = 0.0, temp = 0.0;

  float readn = 0.0;

  float\* mas;

  float\* masC;

  FILE \*file = NULL;

  clock\_t t\_start;

  clock\_t t\_end;

  file = fopen("output\_1.txt", "r");

  if (file == NULL)

  {

    printf("No file named 'output\_1.txt' found. Press any key to exit.");

    getch();

    return 0;

  }

  fscanf(file, "%d", &n);

  mas = (float\*)malloc(n \* sizeof(float));

  masC = (float\*)malloc(n \* sizeof(float));

  for (i = 0; i < n; i++)

  {

    fscanf(file, "%f", &readn);

    mas[i] = readn;

    masC[i] = readn;

  }

  fclose(file);

  while (comm != 3)

  {

    printf("Select command:\n1 - Print numbers\n2 - Sort numbers\n3 - Exit\n");

    scanf("%d", &comm);

    switch(comm)

    {

      case 1:

        {

          for (i = 0; i < n; i++)

          {

            printf("%lf\n", masC[i]);

          }

          printf("\n");

          break;

        }

      case 2:

        {

          t\_start = clock();

          BubbleSort(masC, n);

          t\_end = clock();

          file = fopen("output\_2\_1.txt", "w");

          for (i = 0; i < n; i++)

          {

            fprintf(file, "%lf\n", masC[i]);

            masC[i] = mas[i];

          }

          fclose(file);

          printf("Bubble sort time: %lf seconds.\n", (float)((t\_end -t\_start) / CLOCKS\_PER\_SEC));

          t\_start = clock();

          InsertSort(masC, n);

          t\_end = clock();

          file = fopen("output\_2\_2.txt", "w");

          for (i = 0; i < n; i++)

          {

            fprintf(file, "%lf\n", masC[i]);

            masC[i] = mas[i];

          }

          fclose(file);

          printf("Insert sort time: %lf seconds.\n", (float)((t\_end -t\_start) / CLOCKS\_PER\_SEC));

          t\_start = clock();

          QuickSort(masC, n);

          t\_end = clock();

          file = fopen("output\_2\_3.txt", "w");

          for (i = 0; i < n; i++)

          {

            fprintf(file, "%lf\n", masC[i]);

            masC[i] = mas[i];

          }

          fclose(file);

          printf("Quick sort time: %lf seconds.\n", (float)((t\_end -t\_start) / CLOCKS\_PER\_SEC));

          printf("Results of sorts saved into files 'output\_2\_1.txt', 'output\_2\_2.txt', 'output\_2\_3.txt'.\n\n");

          break;

        }

      default:

        break;

    }

  }

  free(mas);

  free(masC);

  return 0;

}