Основная информация

- Язык программирования для выполнения лабораторных работ и курсового проекта: C/C++. Использование структур данных и функций, таких как *std::sort*(), *std::map*, *std::vector* и др., запрещено.
- Лабораторные работы и курсовой проект нельзя сдавать в группах из нескольких человек.
- Баллы за лабораторную работу выставляются при защите однократно, если лабораторная защищена успешно.
- Крайние сроки для сдачи лабораторных работ указаны в таблице с результатами и файле *deadline*.
- Для каждой лабораторной работы предполагается, что студент может ответить на вопросы по исходному коду и работе алгоритма, это общее условие не прописано в заданиях к лабораторным работам.

В приложении: измерение времени работы алгоритма, генерация псевдослучайных чисел, генерация текстового файла со словами, построение графиков, скрипты *gnuplot* для построения графиков.

Приложение

Измерение времени работы алгоритма

Для измерения времени работы можно использовать функцию *wtime*(), либо другую функцию с подходящей точностью измерения времени. Исходный код показан на листинге 1, использование – на листинге 2. Функция возвращает текущее время в секундах. При измерении времени работы алгоритма необходимо минимизировать влияние на время работы частей программы, не являющихся частью алгоритма. Например, следует избегать измерения времени с вызовом функции *printf()* внутри него. При измерении времени работы сортировки (листинг 2) заполнение массива (*fill_rand()*) не учитывается, поскольку не является частью сортировки.

Для наглядности при демонстрации зависимости шаг должен быть одинаковым, например, от 50000 до 1000000 с шагом 50000 (50000, 100000, 150000…).

Для более точного измерения времени можно измерить время работы функции в цикле *for*, например, 100000 раз, суммируя результат. Тогда для получения среднего времени полученный результат необходимо разделить на 100000. Также можно измерять количество тактов процессора (*rdtsc*). Необходимое количество итераций цикла зависит от требуемой точности результата и времени работы функции. Для лабораторных работ достаточно:

- 0,1-1 с. — 5-10 итераций
- 0,001-0,1 с. — 100-1000 итераций
- Менее 0,001 с. — 500-100000 итераций

Листинг 1. Функция *wtime()*

```
#include <sys/time.h>
double wtime()
{
    struct timeval t;
    gettimeofday(&t, NULL);
    return (double)t.tv_sec + (double)t.tv_usec * 1E-6;
}
```

Листинг 2. Пример использования функции *wtime()*

```
…
int *arr = malloc(sizeof(int) * n);
fill_rand(arr);
double t = wtime();
heap_sort(arr, n);
t = wtime() - t;
…
```

Генерация псевдослучайных чисел

Для генерации равномерно распределенного псевдослучайного числа из интервала *[min, max)* может быть использована функция *getrand(min, max)* (листинг 3). Функция *srand()* используется для создания сида (начальной точки ряда псевдослучайных чисел). Для получения одинаковой последовательности чисел при нескольких запусках программы может быть использован одинаковый сид, например, *srand(123)*. Для получения разной последовательности чисел при каждом запуске программы можно использовать в качестве сида текущее время.

Листинг 3. Функция для генерации псевдослучайных чисел

```
#include <stdlib.h>
int getrand(int min, int max)
{
    return (double)rand() / (RAND_MAX + 1.0) * (max - min) + min;
}
```

Генерация текстового файла со словами

```
import random
import time

def main():
    print("\n============================\n"
          "This program generate <n> random words with generator's seed <seed>\n"
          "from <min lenght> to <max length>, from <min ascii> to <max ascii>\n"
          "and write it into <filename>;\n"
          "if generator's seed equals to 0, then seed is current time\n"
          "============================\n")

    n, min_length, max_length, min_ascii, max_ascii, seed = \
        map(int, input("Input:\n<n> <min length> <max length> <min ascii> <max ascii> <seed>\n").split())

    if (max_length < min_length or max_ascii < min_ascii):
        print("<max lenght> should be less than or equal to <min lenght>\n"
              "<max ascii> should be less than or equal to <min ascii>\n")
        exit(1)
    if (max_length < 1 or min_length < 1 or n < 1 or min_ascii < 32 or max_ascii < 32
         or min_ascii > 126 or max_ascii > 126):
        print("all numbers should be greather than 0;\n"
              "<max ascii> and <min ascii> should be less than 126 and greather than 31;\n")
        exit(1)
    if (seed == 0):
        random.seed(time.time())
    else:
        random.seed(seed)

    if (n > 5000000):
        print("n > 5000000, exit\n")
        exit(1)
    elif (n >= 1000000):
        res = input("n = " + str(n) + ", press <Y> to continue\n")
        if (res != "Y"):
            exit(1)

    filename = input("Input <filename>:\n")

    file = open(filename, 'w')
    for i in range(0, n):
        word_len = random.randint(min_length, max_length)
        word = ""
        for j in range(0, word_len):
            word += chr(random.randint(min_ascii, max_ascii))
        word += '\n'
        file.write(word)
    file.close()
    print("Done\n")
main()
```

Сортировка текстового файла

Перед запуском скрипта нужно убедиться, что файл `OUTPUT_FILE` не существует или не содержит важной информации, поскольку его содержимое будет перезаписано.

Запуск скрипта:

`./script.sh <INPUT_FILE> <OUTPUT_FILE>`

```
#!/bin/sh
INPUT_FILE=$1
OUTPUT_FILE=$2
MINCHARS=6
cat $INPUT_FILE | tr -s '\n' | grep -E ".{$MINCHARS}" | sort | \
uniq > $OUTPUT_FILE
```

Построение графиков

Пример графика показан на рисунке 1:

- Читаемый размер шрифта
- При размещении графика в файлах формата *docx*, *odt* или других, шрифт на графике должен примерно соответствовать шрифту текста
- Присутствует «легенда», в которой прописано значение линий на графике
- Оси подписаны с указанием единиц измерения (*секунда, байт…*)
- Количество делений по оси X соответствует количеству результатов по оси X, количество делений по оси Y достаточное для относительно простой точной оценки положения точки на графике
- Если время выполнения одного алгоритма значительно больше/меньше других, нужно либо сделать несколько графиков: 1 со всеми алгоритмами, а 2-й — без алгоритма со значительным отличием по времени, либо использовать логарифмическую шкалу для построения графика

Инструменты, которые можно использовать для построения графиков: *Microsoft Excel, LibreOffice Calc, gnuplot, python* и др.

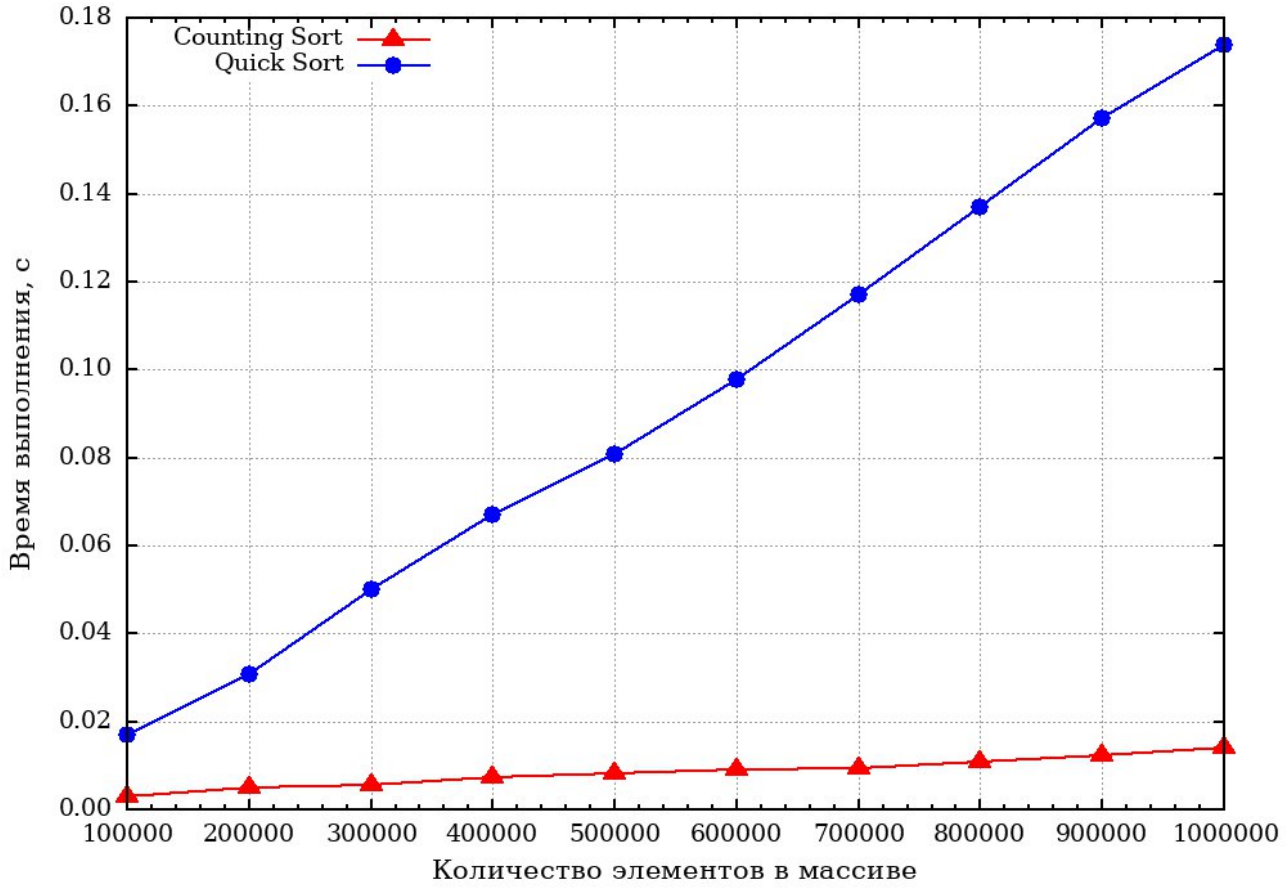

Рисунок 1. Зависимость времени выполнения сортировки подсчетом и быстрой сортировки от количества элементов в массиве

Построение графиков, gnuplot

```
#!/usr/bin/gnuplot
set termoption enhanced
set terminal png size 800,400 font "Times New Roman, 14"
set output "plot123.png"
set style line 1 lc rgb "red" lt 1 lw 2 pt 9 ps 2
set style line 2 lc rgb "blue" lt 1 lw 2 pt 7 ps 2
set style line 3 lc rgb "green" lt 1 lw 2 pt 5 ps 2
set border lw 2
set grid
set key top left
set xlabel "Количество элементов в массиве" font "Times New Roman, 16"
set ylabel "Время выполнения, с" rotate by 90 font "Times New Roman, 16"
set xtics 100000
set mxtics
set rmargin 4
set format x "%6.0f"
set format y "%.2f"
plot "res.dat" using 1:2 title "Counting Sort" with linespoints ls 1, \
"res.dat" using 1:3 title "Quick Sort" with linespoints ls 2, \
```

```
#!/usr/bin/gnuplot
set termoption enhanced
set terminal png size 800,400 font "Times New Roman, 14"
set output "plot123.png"
set style line 1 lc rgb "red" lt 1 lw 2 pt 9 ps 2
set style line 2 lc rgb "blue" lt 1 lw 2 pt 7 ps 2
set style line 3 lc rgb "green" lt 1 lw 2 pt 5 ps 2
set border lw 2
set grid
set key center right
set xlabel "Количество элементов в массиве, тыс" font "Times New Roman, 16"
set ylabel "Время выполнения, мкс" rotate by 90 font "Times New Roman, 16"
set xtics 250 rotate by 90 right
set mxtics
set logscale y 2
set format x "%6.0f"
set format y "%.1f"
plot "t2.txt" using 1:2 title "Linear" with linespoints ls 1, \
"t2.txt" using 1:3 title "Binary" with linespoints ls 2, \
"t2.txt" using 1:4 title "Exponential" with linespoints ls 3, \
```

```
#!/usr/bin/gnuplot
set termoption enhanced
set terminal png size 800,400 font "Times New Roman, 14"
set output "plot1.png"
set style line 1 lc rgb "red" lt 1 lw 2 pt 9 ps 2
set style line 2 lc rgb "blue" lt 1 lw 2 pt 7 ps 2
set style line 3 lc rgb "green" lt 1 lw 2 pt 5 ps 2
set border lw 2
set grid
set key top left
set xlabel "Количество элементов в массиве, тыс" font "Times New Roman, 16"
set ylabel "Время выполнения, мкс" rotate by 90 font "Times New Roman, 16"
set xtics 250 rotate by 90 right
set mxtics
set format x "%6.0f"
set format y "%.2f"
plot "t2.txt" using 1:3 title "Binary" with linespoints ls 1, \
"t2.txt" using 1:4 title "Exponential" with linespoints ls 2, \
```