Московский Авиационный Институт

(Национальный Исследовательский Университет)

**Факультет прикладной математики и физики**

Кафедра вычислительной математики и программирования

**КУРСОВАЯ РАБОТА**

**по курсу**

**“Практикум на ЭВМ”**

**II семестр**

**«Сортировка и поиск»**

Студент: Бокоч С.М.

Группа: 08-104, № по списку 2

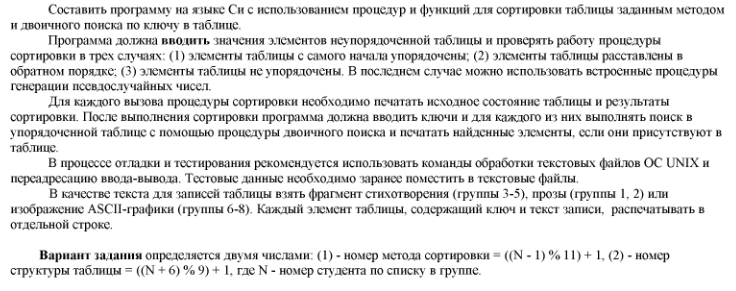
Руководитель: Никулин С.П.,  
 доцент каф.806

Оценка:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Москва, 2017**

**1.Задание**

**2. Общий метод решения**

* Вводим последовательно данные из файла: ключ — значение. Комплексное число мы должны преобразовать в элемент структуры из строки, а обработанное значение может содеражть пробелы между словами.
* Печать таблицы очевидна: преобразовываем комплексное число обратно в строку.
* Бинарный поиск осуществляется толь в отстортированном массиве, поэтому проверяем его на отсортированность перед поиском
* В сортировке подсчетом. Вообще говоря, нельзя сравнивать комплексные числа, но можно сортировать по ключи по модулю. Для сортивки заведем «вспомогательный» массив структуры и массив count для нахождения индекса, и подсчетом считаем количество элементов для i-ого, ключи которых по модулю больше текущего в массиве count. Получим, что массив count содежит индексы элементов сортируемого массива. После делаем обмен.
* Перемешивание осуществим с помощью метода rand() из стандартной библиотеки C.
* Реверс с помощью линейного обмена крайних элемнтов в цикле

**3. Общие сведения о программe**

Необходимое программное и аппаратное обеспечение: компилятор gcc

Операционная система: любая операционная система с поддержкой Си

Язык: Си

Система программирования: Си

Число строк программ:

* main.c - 243

Местонахождение и имена файлов с исходными текстами и данными: serega@serega-Inspiron-3537:~/course\_project/kp9

Способ вызова и загрузки: в директории с файлом в bash ./main

**4. Функциональное назначение**

Программа расчитанна на считывание элементов, количество которых указано в программе программистом. В данном случае 100;

Длина значения не должно превышать 50 символов;

Ключ должен иметь тип a+b\*i, где a и b целые числа, могут быть не заданы. Пример: a, b\*i, « » ( нулевое значение )

**6. Описание переменных и констант**

**Файл main.c**

|  |  |
| --- | --- |
| #define simb ((ch >= 48 && ch <= 57) || (ch >= 65 && ch <= 90) || (ch >= 97 && ch <= 122)) | Проверка на корректных ввод |
| #define N 100 | Количество элементов |
| typedef struct \_comp  {  int a;  int b;  } comp; | Структура комплексного числа |
| typedef struct \_temp  {  comp \_key;  char \*\_data;  } temp; | Элемент сортируемой структуры |
| int size\_arr(char c[])  {  int i = 0;  while (c[i] != '\0')  i++;  return i;  }  int toInt(char c[], int \*index)  {  int a = 0;  int i = \*index;  int p = 1;  while (c[i] != '+' && c[i] != '-')  {  if (c[i] >= '0' && c[i] <= '9')  {  a += p \* (c[i] - 48);  p \*= 10;  }  i--;  if (i < 0)  break;  }  if (i >= 0)  {  if (c[i] == '-')  a = -a;  }  \*index = i;  return a;  }  int modComp(comp x)  {  return x.a \* x.a + x.b \* x.b;  } | Возвращает длину строки  Специальный перевод строки в тип int  Годен для обработки комплексного числа.  Находит модуль комплексного числа |
| comp ReadKey(char c[])  {  comp t;  t.a = 0;  t.b = 0;  /\* Читаем первую цифру, добавляем в мнимую часть \*/  int length = size\_arr(c);  for (int i = length - 1; i >= 0; i--)  {  if (c[i] == 'i')  t.b = toInt(c, &i);  else  t.a = toInt(c, &i);  }  return t;  }  char \*ReadData(FILE \*stream)  {  char \*str = NULL;  char ch = '\0';  int i = 0;  int y = 0;  while (!simb)  {  ch = fgetc(stream);  }  do  {  str = (char \*)realloc(str, sizeof(char) \* (i + 1));  str[i] = ch;  ch = fgetc(stream);  i++;  } while (simb || ch == ' ');  str = (char \*)realloc(str, sizeof(char) \* (i + 1));  str[i] = '\0';  return str;}  void CompToStr(comp x, char ch[])  {  int n = 0;  if (x.a != 0)  sprintf(ch, "%d", x.a);  if (x.b != 0)  {  if (x.a != 0)  n = size\_arr(ch);  if (x.b > 0)  sprintf(ch + n, "+%di", x.b);  else  sprintf(ch + n, "%di", x.b);  }  if (x.a == x.b && x.a == 0)  sprintf(ch, "%d", 0);  return;}  void swapTemp(temp \*a, temp \*b)  {  temp t = \*a; \*a = \*b; \*b = t;  return;  } | Обработчик строки комплексного числа.  Возврашает обработанное комплесное число в виде структуры.  Вводим строку — значение, допустимо с пробелами.  Переводит комплексное число в строку  Обмен элементов. |
| void printTable(temp a[], const int size)  {  printf("+---------+------------------------------------------------+\n");  printf("| Ключ | Значение |\n");  printf("+---------+------------------------------------------------+\n");  for (int i = 0; i < size; i++)  {  char c[10];  CompToStr(a[i].\_key, c);  printf("|%4s |%48s|\n", c, a[i].\_data);  }  printf("+---------+------------------------------------------------+\n");  } | Печать таблицы |
| void reverse(temp a[], int size)  {  int i, j;  for (int i = 0; i < size / 2; i++)  swapTemp(&a[i], &a[size - i - 1]);  return;  }  void scramble(temp \*a, const int size)  {  int i, j, k;  srand((unsigned int)time(0));  for (k = 0; k < size; k++)  {  i = rand() % size;  j = rand() % size;  swapTemp(&a[i], &a[j]);  }  } | Выводит элементы структуры в обратном порядке.  Перемешивает элементы структуры |
| int isSorted(temp a[], int size)  {  for (int i = 0; i < size - 1; i++)  {  if (modComp(a[i].\_key) > modComp(a[i + 1].\_key))  return 0;  }  return 1;  }  int binSearch(temp a[], int n, temp x)  {  int lhs = 0, rhs = n - 1, mid;  if (rhs <= 0)  return -1;  int xMod = modComp(x.\_key);  while (lhs < rhs)  {  mid = (lhs + rhs) / 2;  int midMod = modComp(a[mid].\_key);  if (xMod == midMod)  return mid;  else if (xMod > midMod)  lhs = mid + 1;  else  rhs = mid;  }  if (modComp(a[rhs].\_key) == xMod)  return rhs;  return -1;  }  void sort(temp a[], const int size)  {  int count[size];  temp b[size];  for (int i = 0; i < size; i++)  count[i] = 0;  for (int i = 0; i < size - 1; i++)  for (int j = i + 1; j < size; j++)  if (modComp(a[i].\_key) > modComp(a[j].\_key))  count[i]++;  else  count[j]++;  for (int i = 0; i < size; i++)  b[count[i]] = a[i];  for (int i = 0; i < size; i++)  a[i] = b[i];  return;  } | Проверяет структуру на отсортированность  Бинарный поиск.  Возвращает -1 если элемент не найден, иначе выдает позицию найденного элемента.  Сортировка подсчетом. |
| int main()  {  FILE \*in = fopen("test.txt", "r");  if (in == NULL)  {  printf("Неудается открыть файл\n");  return 1;  }  temp arr[N];  temp tmp;  int index = 0;  char key[10];  /\* Enter DATA \*/  while (fscanf(in, "%s", key) == 1)  {  comp C = ReadKey(key);  char \*data = ReadData(in);  tmp.\_key = C;  tmp.\_data = data;  arr[index] = tmp;  index++;  }  int SIZE = index;  fclose(in);  /\* end enter data \*/  int action;  do  {  printf("Меню\n");  printf("1) Печать\n");  printf("2) Двоичный поиск\n");  printf("3) Сортировка\n");  printf("4) Перемешивание\n");  printf("5) Реверс\n");  printf("6) Выход\n");  printf("Выберите действие\n");  scanf("%d", &action);  switch (action)  {  case 1:  {  printTable(arr, SIZE);  }  break;  case 2:  {  if (!isSorted(arr, SIZE))  printf("Ошибка. Таблица не отсортирована\n");  else  {  printf("Введите ключ: ");  scanf("%s", key);  tmp.\_key = ReadKey(key);  int i = binSearch(arr, SIZE, tmp);  if (i > -1)  printf("Найдена строка: %s\n", arr[i].\_data);  else  printf("Строка с таким ключом не найдена\n");  }  }  break;  case 3:  {  sort(arr, SIZE);  }  break;  case 4:  {  scramble(arr, SIZE);  }  break;  case 5:  {  reverse(arr, SIZE);  }  break;  case 6:  break;  default:  {  printf("Ошибка. Такого пункта меню не существует\n");  }  break;  }  } while (action != 6);  return 0;  } |  |

**Тестовые примеры:**

serega@serega-Inspiron-3537:~/course\_project/kp9$ cat test.txt

5+6\*i Mac Kenlly

4+7\*i ROCK in your life

2-6\*i Gerbet

6\*i Bokoch

-9 Nikulin

10 Crazy FROG

0 Zaicev

serega@serega-Inspiron-3537:~/course\_project/kp9$ gcc main.c

serega@serega-Inspiron-3537:~/course\_project/kp9$ ./a.out

Меню

1) Печать

2) Двоичный поиск

3) Сортировка

4) Перемешивание

5) Реверс

6) Выход

Выберите действие

1

+---------+------------------------------------------------+

| Ключ | Значение |

+---------+------------------------------------------------+

|5+6i | Mac Kenlly|

|4+7i | ROCK in your life|

|2-6i | Gerbet|

| +6i | Bokoch|

| -9 | Nikulin|

| 10 | Crazy FROG|

| 0 | Zaicev|

+---------+------------------------------------------------+

Меню

1) Печать

2) Двоичный поиск

3) Сортировка

4) Перемешивание

5) Реверс

6) Выход

Выберите действие

2

Ошибка. Таблица не отсортирована

Меню

1) Печать

2) Двоичный поиск

3) Сортировка

4) Перемешивание

5) Реверс

6) Выход

Выберите действие

3

Меню

1) Печать

2) Двоичный поиск

3) Сортировка

4) Перемешивание

5) Реверс

6) Выход

Выберите действие

1

+---------+------------------------------------------------+

| Ключ | Значение |

+---------+------------------------------------------------+

| 0 | Zaicev|

| +6i | Bokoch|

|2-6i | Gerbet|

|5+6i | Mac Kenlly|

|4+7i | ROCK in your life|

| -9 | Nikulin|

| 10 | Crazy FROG|

+---------+------------------------------------------------+

Меню

1) Печать

2) Двоичный поиск

3) Сортировка

4) Перемешивание

5) Реверс

6) Выход

Выберите действие

4

Меню

1) Печать

2) Двоичный поиск

3) Сортировка

4) Перемешивание

5) Реверс

6) Выход

Выберите действие

1

+---------+------------------------------------------------+

| Ключ | Значение |

+---------+------------------------------------------------+

|2-6i | Gerbet|

| -9 | Nikulin|

| +6i | Bokoch|

| 0 | Zaicev|

|5+6i | Mac Kenlly|

|4+7i | ROCK in your life|

| 10 | Crazy FROG|

+---------+------------------------------------------------+

Меню

1) Печать

2) Двоичный поиск

3) Сортировка

4) Перемешивание

5) Реверс

6) Выход

Выберите действие

3

Меню

1) Печать

2) Двоичный поиск

3) Сортировка

4) Перемешивание

5) Реверс

6) Выход

Выберите действие

1

+---------+------------------------------------------------+

| Ключ | Значение |

+---------+------------------------------------------------+

| 0 | Zaicev|

| +6i | Bokoch|

|2-6i | Gerbet|

|5+6i | Mac Kenlly|

|4+7i | ROCK in your life|

| -9 | Nikulin|

| 10 | Crazy FROG|

+---------+------------------------------------------------+

Меню

1) Печать

2) Двоичный поиск

3) Сортировка

4) Перемешивание

5) Реверс

6) Выход

Выберите действие

4

Меню

1) Печать

2) Двоичный поиск

3) Сортировка

4) Перемешивание

5) Реверс

6) Выход

Выберите действие

1

+---------+------------------------------------------------+

| Ключ | Значение |

+---------+------------------------------------------------+

| -9 | Nikulin|

| +6i | Bokoch|

| 0 | Zaicev|

|5+6i | Mac Kenlly|

|2-6i | Gerbet|

|4+7i | ROCK in your life|

| 10 | Crazy FROG|

+---------+------------------------------------------------+

Меню

1) Печать

2) Двоичный поиск

3) Сортировка

4) Перемешивание

5) Реверс

6) Выход

Выберите действие

3

Меню

1) Печать

2) Двоичный поиск

3) Сортировка

4) Перемешивание

5) Реверс

6) Выход

Выберите действие

5

Меню

1) Печать

2) Двоичный поиск

3) Сортировка

4) Перемешивание

5) Реверс

6) Выход

Выберите действие

1

+---------+------------------------------------------------+

| Ключ | Значение |

+---------+------------------------------------------------+

| 10 | Crazy FROG|

| -9 | Nikulin|

|4+7i | ROCK in your life|

|5+6i | Mac Kenlly|

|2-6i | Gerbet|

| +6i | Bokoch|

| 0 | Zaicev|

+---------+------------------------------------------------+

Меню

1) Печать

2) Двоичный поиск

3) Сортировка

4) Перемешивание

5) Реверс

6) Выход

Выберите действие

5

Меню

1) Печать

2) Двоичный поиск

3) Сортировка

4) Перемешивание

5) Реверс

6) Выход

Выберите действие

1

+---------+------------------------------------------------+

| Ключ | Значение |

+---------+------------------------------------------------+

| 0 | Zaicev|

| +6i | Bokoch|

|2-6i | Gerbet|

|5+6i | Mac Kenlly|

|4+7i | ROCK in your life|

| -9 | Nikulin|

| 10 | Crazy FROG|

+---------+------------------------------------------------+

**9. Дневник отладки**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Дата | Время | Место | Наиболее характерные ошибки | Действия по исправлению | Внешние признаки | Сведения о степени самостоятель-ности |
| 1 | 29.05.2017 | 17:01. | Общежитие №5 МАИ | Зацикливание при считывании строки без пробеллов | Добавить проверку провильности считываемых символов. Это буквы строчные и прописные, цифры и пробел. | Зацикливание | Обнаружено самосто-ятельно |

*Примечание:* sprintf(str, «%d», DEC); - записывает в массив символов десятичный вид числа

**12. Выводы по задаче**

Проделав данную лабораторную работу, я изучил различные методы сортировки, научился использовать их на практике. Сортировка по ключу очень полезный и надежный етода

***Протокол исходного кода***

#include "stdio.h"

#include "stdlib.h"

#include "string.h"

#include "malloc.h"

#include "string.h"

#include "time.h"

#define simb ((ch >= 48 && ch <= 57) || (ch >= 65 && ch <= 90) || (ch >= 97 && ch <= 122))

#define N 100

/\* ключ комлексный \*/

/\* сортировка подсчетом \*/

typedef struct \_comp

{

int a;

int b;

} comp;

typedef struct \_temp

{

comp \_key;

char \*\_data;

} temp;

int size\_arr(char c[])

{

int i = 0;

while (c[i] != '\0')

i++;

return i;

}

int toInt(char c[], int \*index)

{

int a = 0;

int i = \*index;

int p = 1;

while (c[i] != '+' && c[i] != '-')

{

if (c[i] >= '0' && c[i] <= '9')

{

a += p \* (c[i] - 48);

p \*= 10;

}

i--;

if (i < 0)

break;

}

if (i >= 0)

{

if (c[i] == '-')

a = -a;

}

\*index = i;

return a;

}

comp ReadKey(char c[])

{

comp t;

t.a = 0;

t.b = 0;

/\* Читаем первую цифру, добавляем в мнимую часть \*/

int length = size\_arr(c);

for (int i = length - 1; i >= 0; i--)

{

if (c[i] == 'i')

t.b = toInt(c, &i);

else

t.a = toInt(c, &i);

}

return t;

}

char \*ReadData(FILE \*stream)

{

char \*str = NULL;

char ch = '\0';

int i = 0;

int y = 0;

while (!simb)

{

ch = fgetc(stream);

}

do

{

str = (char \*)realloc(str, sizeof(char) \* (i + 1));

str[i] = ch;

ch = fgetc(stream);

i++;

} while (simb || ch == ' ');

str = (char \*)realloc(str, sizeof(char) \* (i + 1));

str[i] = '\0';

return str;

}

int modComp(comp x)

{

return x.a \* x.a + x.b \* x.b;

}

int isSorted(temp a[], int size)

{

for (int i = 0; i < size - 1; i++)

{

if (modComp(a[i].\_key) > modComp(a[i + 1].\_key))

return 0;

}

return 1;

}

int binSearch(temp a[], int n, temp x)

{

int lhs = 0, rhs = n - 1, mid;

if (rhs <= 0)

return -1;

int xMod = modComp(x.\_key);

while (lhs < rhs)

{

mid = (lhs + rhs) / 2;

int midMod = modComp(a[mid].\_key);

if (xMod == midMod)

return mid;

else if (xMod > midMod)

lhs = mid + 1;

else

rhs = mid;

}

if (modComp(a[rhs].\_key) == xMod)

return rhs;

return -1;

}

void swapTemp(temp \*a, temp \*b)

{

temp t = \*a;

\*a = \*b;

\*b = t;

return;

}

void scramble(temp \*a, const int size)

{

int i, j, k;

srand((unsigned int)time(0));

for (k = 0; k < size; k++)

{

i = rand() % size;

j = rand() % size;

swapTemp(&a[i], &a[j]);

}

}

void CompToStr(comp x, char ch[])

{

int n = 0;

if (x.a != 0)

sprintf(ch, "%d", x.a);

if (x.b != 0)

{

if (x.a != 0)

n = size\_arr(ch);

if (x.b > 0)

sprintf(ch + n, "+%di", x.b);

else

sprintf(ch + n, "%di", x.b);

}

if (x.a == x.b && x.a == 0)

sprintf(ch, "%d", 0);

return;

}

void printTable(temp a[], const int size)

{

printf("+---------+------------------------------------------------+\n");

printf("| Ключ | Значение |\n");

printf("+---------+------------------------------------------------+\n");

for (int i = 0; i < size; i++)

{

char c[10];

CompToStr(a[i].\_key, c);

printf("|%4s |%48s|\n", c, a[i].\_data);

}

printf("+---------+------------------------------------------------+\n");

}

void reverse(temp a[], int size)

{

int i, j;

for (int i = 0; i < size / 2; i++)

swapTemp(&a[i], &a[size - i - 1]);

return;

}

void sort(temp a[], const int size)

{

int count[size];

temp b[size];

for (int i = 0; i < size; i++)

count[i] = 0;

for (int i = 0; i < size - 1; i++)

for (int j = i + 1; j < size; j++)

if (modComp(a[i].\_key) > modComp(a[j].\_key))

count[i]++;

else

count[j]++;

for (int i = 0; i < size; i++)

b[count[i]] = a[i];

for (int i = 0; i < size; i++)

a[i] = b[i];

return;

}

int main()

{

FILE \*in = fopen("test.txt", "r");

if (in == NULL)

{

printf("Неудается открыть файл\n");

return 1;

}

temp arr[N];

temp tmp;

int index = 0;

char key[10];

/\* Enter DATA \*/

while (fscanf(in, "%s", key) == 1)

{

comp C = ReadKey(key);

char \*data = ReadData(in);

tmp.\_key = C;

tmp.\_data = data;

arr[index] = tmp;

index++;

}

int SIZE = index;

fclose(in);

/\* end enter data \*/

int action;

do

{

printf("Меню\n");

printf("1) Печать\n");

printf("2) Двоичный поиск\n");

printf("3) Сортировка\n");

printf("4) Перемешивание\n");

printf("5) Реверс\n");

printf("6) Выход\n");

printf("Выберите действие\n");

scanf("%d", &action);

switch (action)

{

case 1:

{

printTable(arr, SIZE);

}

break;

case 2:

{

if (!isSorted(arr, SIZE))

printf("Ошибка. Таблица не отсортирована\n");

else

{

printf("Введите ключ: ");

scanf("%s", key);

tmp.\_key = ReadKey(key);

int i = binSearch(arr, SIZE, tmp);

if (i > -1)

printf("Найдена строка: %s\n", arr[i].\_data);

else

printf("Строка с таким ключом не найдена\n");

}

}

break;

case 3:

{

sort(arr, SIZE);

}

break;

case 4:

{

scramble(arr, SIZE);

}

break;

case 5:

{

reverse(arr, SIZE);

}

break;

case 6:

break;

default:

{

printf("Ошибка. Такого пункта меню не существует\n");

}

break;

}

} while (action != 6);

return 0;

}