Задача 1

Полностью объяснить задачу из второго задания № 7. Под «полностью объяснить» подразумевается все — от свойств используемых структур данных до строчек кода.

Формат сдачи — предоставленный исходный код плюс устная беседа по нему.

Задача 2

Разобрать алгоритм сортировки массива выбором (см. ниже). Оценить асимптотическую сложность алгоритма.

Формат сдачи — названная асимптотическая сложность (как «О большое от чего-то») плюс устная беседа по алгоритму.

Задача 3

На базе любой удобной задачи второго задания реализовать структуру данных и интерфейс к ней (указаны ниже). Программа при запуске должна читать команды из файла commands.txt, выполнять их, печатать результат на экран и завершаться. Никаких интерактивных действий от пользователя не предполагается.

Структура данных: односвязный список

Тип данных: строка не более 15 символов

Состав инструкций в commands.txt:

INSERT A — добавить элемент со значением A, ничего не печатать. Если такой элемент уже есть — добавить еще одну копию.

FIND A — найти элемент со значением A. Если элемент найден, напечатать FOUND. Если не найден — напечатать NOT FOUND.

DELETE A — найти и удалить элемент со значением A. Если элемента нет — ничего не делать. Если элементов несколько — удалить одну из копий.

PRINT_INVERSE — распечатать список от последнего элемента к первому, сам список при этом измениться не должен

Пример:

Инструкции в файле:

INSERT alice

INSERT bob

DELETE alice

FIND alice

Вывод программы:

NOT FOUND (так как alice уже удалено)

Формат сдачи — предоставленный исходный код плюс устная беседа по нему.

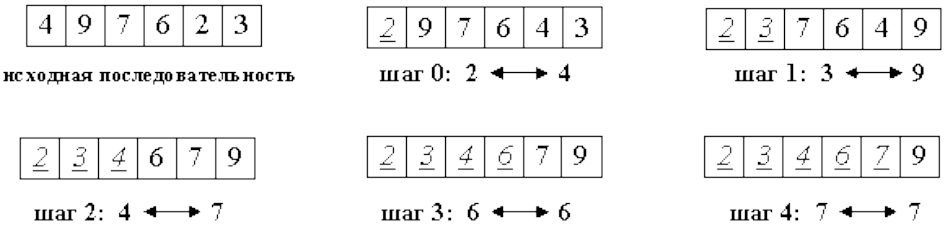
Сортировка выбором

Задача — отсортировать массив.

Идея метода состоит в том, чтобы создавать отсортированную последовательность путем присоединения к ней одного элемента за другим в правильном порядке.

Будем строить готовую последовательность, начиная с левого конца массива. Алгоритм состоит из n последовательных шагов, начиная от нулевого и заканчивая (n-1)-м.

На i-м шаге выбираем наименьший из элементов a[i] ... a[n] и меняем его местами с a[i]. Последовательность шагов при n=5 изображена на рисунке ниже.



Вне зависимости от номера текущего шага i, последовательность a[0]...a[i] (выделена курсивом) является упорядоченной. Таким образом, на (n-1)-м шаге вся последовательность, кроме a[n] оказывается отсортированной, а a[n] стоит на последнем месте по праву: все меньшие элементы уже ушли влево.

Пример реализации:

```
template<class T>
void selectSort(T a[], long size) {
  long i, j, k;
  T x;
  for( i=0; i < size; i++) {    // i - номер текущего шага
    k=i; x=a[i];
    for( j=i+1; j < size; j++)  // цикл выбора наименьшего элемента
      if (  a[j] < x ) {
        k=j; x=a[j];            // k - индекс наименьшего элемента
      }
    a[k] = a[i]; a[i] = x;      // меняем местами наименьший с a[i]
  }
}
```