26 (высокий уровень, время – 35 минут)

Тема: Обработка массива целых чисел из файла. Сортировка.

Что проверяется:

Умение обрабатывать целочисленную информацию с использованием сортировки.

1.6.3. Построение алгоритмов и практические вычисления.

1.1.3. Строить информационные модели объектов, систем и процессов в виде алгоритмов.

Что нужно знать при написании программы:

Чтение данных из файла

- в языке Python для чтения данных удобно использовать менеджер контекста, который открывает файл и закрывает его; например, код

```
with open("26.txt") as Fin:
  ... # какие-то операции с файлом
```

равносилен такому

```
Fin = open("26.txt")
... # какие-то операции с файлом
Fin.close()
```

- если в текущей строке файла находится целое число, то прочитать его в переменную **x** можно так можно так:

```
x = int( Fin.readline() )
```

- если в строке записаны два числа, после чтения (**Fin.readline()**) строку нужно разбить на отдельные части по пробелам (каждая часть – символьная запись числа) и затем каждую часть преобразовать в целое число; например, чтение двух чисел:

```
s = Fin.readline()
symData = s.split()
x, y = map( int, symData )
```

или в компактной форме

```
x, y = map( int, Fin.readline().split() )
```

- в языке PascalABC.NET для чтения данных проще всего просто перенаправить входной поток на файл:

```
Assign( input, '26.txt' );
```

после этого можно использовать операторы **read** и **readln**, так же, как при вводе с клавиатуры

- в языке C++ можно читать данные с помощью входного потока (**fstream**):

```
#include <fstream>
...
ifstream Fin("26.txt");
Fin >> x;
Fin >> y >> z;
```

Хранение массива данных

- в языке Python для хранения массива данных используется список; следующая программа показывает чтение массива данных размера N в список **data** из файла «26.txt» (данные записаны в столбик):

```
data = [0]*N
with open("26.txt") as Fin:
  for i in range(N):
    data[i] = int( Fin.readline() )
```

- в языке PascalABC.NET используем динамический массив; когда станет известен его размер, выделаем место в памяти и читаем из входного потока:

```
var data: array of integer;
SetLength( data, N );
for var i:=0 to N-1 do
  read( data[i] );
```

- в языке C++ аналогично используется коллекция **vector**:

```
#include <vector>
...
vector <int> data(N);
for( int i = 0; i < N; i++ )
  Fin >> data[i];
```

Пример задания:

P-00 (демо-2021). *Системный администратор раз в неделю создаёт архив пользовательских файлов. Однако объём диска, куда он помещает архив, может быть меньше, чем суммарный объём архивируемых файлов. Известно, какой объём занимает файл каждого пользователя. По заданной информации об объёме файлов пользователей и свободном объёме на архивном диске определите максимальное число пользователей, чьи файлы можно сохранить в архиве, а также максимальный размер имеющегося файла, который может быть сохранён в архиве, при условии, что сохранены файлы максимально возможного числа пользователей.*

***Входные данные**. В первой строке входного файла **26.txt** находятся два числа: S – размер свободного места на диске (натуральное число, не превышающее 10 000) и N – количество пользователей (натуральное число, не превышающее 1000). В следующих N строках находятся значения объёмов файлов каждого пользователя (все числа натуральные, не превышающие 100), каждое в отдельной строке. Запишите в ответе два числа: сначала наибольшее число пользователей, чьи файлы могут быть помещены в архив, затем максимальный размер имеющегося файла, который может быть сохранён в архиве, при условии, что сохранены файлы максимально возможного числа пользователей.*

Пример входного файла:

```
100 4
80
30
50
40
```

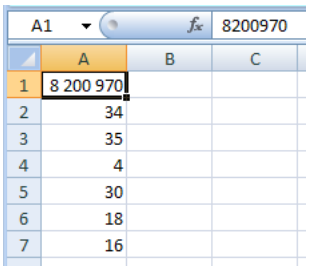
При таких исходных данных можно сохранить файлы максимум двух пользователей. Возможные объёмы этих двух файлов 30 и 40, 30 и 50 или 40 и 50. Наибольший объём файла из перечисленных пар – 50, поэтому ответ для приведённого примера:

```
2 50
```

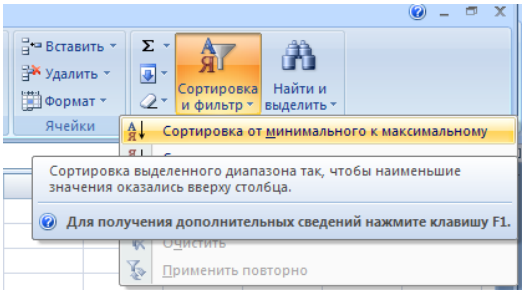
Решение (электронные таблицы, А. Сидоров, www.youtube.com/watch?v=LwTZAHsno0k):

1) сначала загружаем данные в электронную таблицу; тут есть два способа:
   а) открыть файл в Блокноте, выделить всё с помощью клавиш Ctrl+A, скопировать в буфер обмена, а затем вставить в электронную таблицу
   б) выбрать пункт меню *Файл – Открыть*, в окне выбора файла выбрать «Все файлы», и затем выбрать нужный файл; при этом запустится мастер импорта, который загрузит данные в таблицу (он задаст пару вопросов, можно везде нажимать кнопку *Далее*)

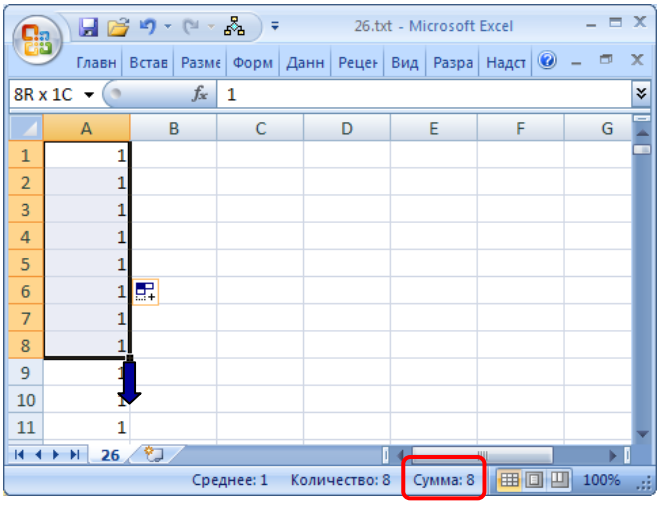
вот что должно получиться:



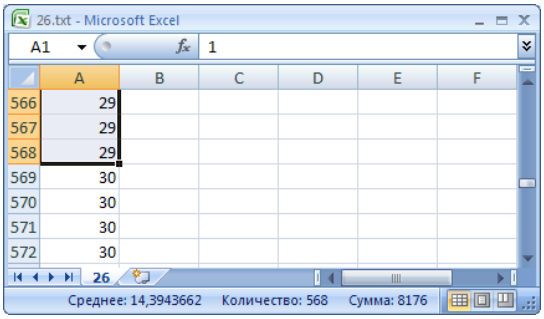
2) для удобства хочется удалить первую строку, которая содержит не такие данные, как все остальные;
3) из первой строки нам нужно число 8200 – это размер свободного места на диске; это число нужно запомнить или где-то записать на бумажке или в ячейке электронной таблицы, только не в первой строке, которая будет удалена); затем удаляем первую строку
4) для того чтобы разместить наибольшее количество файлов, нужно начинать с самых маленьких, то есть данные в первом столбце нужно отсортировать по возрастанию; щёлкнем по заголовку столбца A, чтобы выделить его, от отсортируем:



5) далее начинаем выделять ячейки первого столбца, отслеживая значение суммы в строке состояния

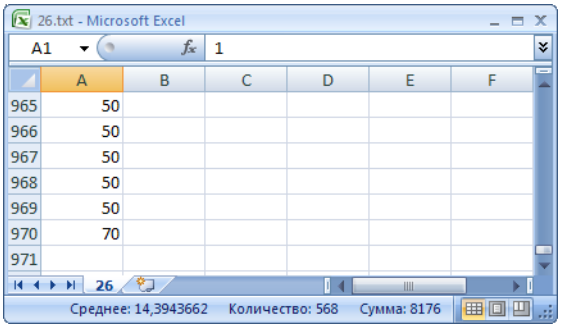


6) нужно выделить наибольшее количество данных, сумма которых не больше, чем 8200:



7) если (см. рисунок) выделить еще одно число, сумма 8206 уже превысит 8200, что недопустимо; поэтому первый ответ – 568

8) у нас есть запас Δ = 8200 – 8176 (текущая сумма) = 24
9) для того чтобы ещё увеличить сумму (но сохранить её не превышающей 8200) мы заменим одно из выделенных значений другим, бóльшим;
10) докажем, что нужно заменять именно самое большое из выбранных значений; пусть мы заменяем значение, равное x, бóльшим значением, равным X; чтобы сумма не превысила заданную, их разница не должна быть больше, чем Δ, то есть $X \le x + \Delta$; таким образом, при бóльшем x мы можем взять бóльшее X
11) наш запас 8200 – 8176 (текущая сумма) = 24, поэтому вместо самого большого взятого значения 29 можно добавить 29 + 24 = 53
12) смотрим в таблицу – значения 53 у нас нет, сразу за 50 идёт 70:



13) поэтому второй ответ – 50
14) Ответ: 568 50

Решение (программа):

1) проще всего составить программу на языке Python, где есть много встроенных функций
2) сначала нужно прочитать данные из файла; читаем все строки сразу в массив **data**:
```
with open("26.txt") as Fin:
  data = Fin.readlines()
```
3) декодируем два числа из первой строки; первой записываем в переменную S, а второе нам не интересно, записываем его в переменную с именем «_»; первую строку сразу удаляем
```
S, _ = map( int, data[0].split() )
del data[0]
```
4) преобразуем данные в целые числа и сразу сортируем
```
data = sorted( list(map(int, data)) )
```
5) теперь накапливаем сумму в переменной **total**, пока она остается не больше, чем **S**:
```
total = 0
for i, val in enumerate(data):
  if total + val > S: break
  total += val
```
6) как только сумма превысила **S**, произойдёт выход из цикла по оператору **break**, а в переменной **i** останется количество добавленных значений; выводим его на экран:
```
print(i)
```
7) вычисляем запас, который мы можем уменьшить с помощью замены одного выбранного значения на другое:
```
delta = S - total
```
8) теперь выбираем из массива данных те значения, которые могут быть выбраны: разность между таким значением и наибольшим выбранным элементом data[i] должна быть не больше, чем **delta**:
```
candidates = [x for x in data
                  if x-data[i] <= delta]
```
9) остается найти второй ответ: максимум из чисел-кандидатов:

```
print( max(candidates) )
```

10) приведём полную программу

```
with open("26.txt") as Fin:
  data = Fin.readlines()
S, _ = map( int, data[0].split() )
del data[0]
data = sorted( list(map(int, data)) )

total = 0
for i, val in enumerate(data):
  if total + val > S: break
  total += val
print(i)

delta = S - total
candidates = [x for x in data
                if x-data[i] <= delta]
print( max(candidates) )
```

11) аналогичная программа на языке **PascalABC.NET**:

```
var S, N, count: integer;
    data: array of integer;
begin
  Assign( input, '26.txt' );
  readln( S, N );
  SetLength( data, N );
  for var i:=0 to N-1 do
    read( data[i] );
  Sort( data );
  var total := 0;
  for count:=0 to N-1 do begin
    if total + data[count] > S then break;
    total += data[count];
  end;
  var delta := S - total;
  var candidates := data.Where( x -> x - data[count] <= delta );
  Println( count, candidates.Max )
end.
```

12) аналогичная программа на языке **C++**:

```
#include <iostream>
#include <fstream>
#include <vector>
#include <algorithm>.

using namespace std;
int main()
{
  ifstream Fin("26.txt");
  int S, N, x;

  Fin >> S >> N;
  vector <int> data(N);
```

```
  for( int i = 0; i < N; i++ )
       Fin >> data[i];

  sort( data.begin(), data.end() );

  int total = 0, count;
  for( count = 0; count < N; count++ ) {
    if( total + data[count] > S ) break;
    total += data[count];
    }

  int delta = S - total;
  int maxCandidate = 0;
  for( int i = count; i < N; i++ ) {
    if( data[i] - data[count] <= delta )
      if( data[i] > maxCandidate )
        maxCandidate = data[i];
    }

  cout << count << " " << maxCandidate;
}
```

Задачи для тренировки:

1) В текстовом файле **27-1.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
2) В текстовом файле **27-2.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
3) В текстовом файле **27-3.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
4) В текстовом файле **27-4.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
5) В текстовом файле **27-5.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
6) В текстовом файле **27-6.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
7) В текстовом файле **27-7.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
8) В текстовом файле **27-8.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
9) В текстовом файле **27-9.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
10) В текстовом файле **27-10.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
11) В текстовом файле **27-11.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
12) В текстовом файле **27-12.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
13) В текстовом файле **27-13.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
14) В текстовом файле **27-14.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
15) В текстовом файле **27-15.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
16) В текстовом файле **27-16.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
17) В текстовом файле **27-17.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
18) В текстовом файле **27-18.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
19) В текстовом файле **27-19.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.
20) В текстовом файле **27-20.txt** находятся данные в формате, описанном выше в формулировке задачи P00. Решите задачу P00.