24 (высокий уровень, время – 18 минут)

Тема: Обработка символьных строк

Что проверяется:

Умение создавать собственные программы (10–20 строк) для обработки символьной информации.

1.5.2. Цепочки (конечные последовательности), деревья, списки, графы, матрицы (массивы), псевдослучайные последовательности.

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

Что нужно знать:

- сначала нужно прочитать строку из файла; эта задача в разных языках программирования решается несколько по-разному
- в языке Python удобнее всего использовать такую конструкцию:

```
with open("k7.txt", "r") as F:
  s = F.readline()
```

конструкция **with-as** – это *контекстный менеджер*, в данном случае он открывает указанный файл в режиме чтения (второй аргумент «**r**» при вызове функции **open**), записывает ссылку на него в файловую переменную **F**, выполняет тело блока (читает первую строку файла в переменную **s**) и закрывает (освобождает) файл

- в языке PascalABC.NET можно выполнить перенаправление потока ввода:

```
assign( input, 'k7.txt' );
readln(s);
```

программа будет «думать», что читает данные, введённые с клавиатуры (с консоли), а на самом деле эти данные будут прочитаны из файла **k7.txt**

- в языке FreePascal также можно выполнить перенаправление потока ввода, но нужно дополнительно открывать входной поток:

```
assign( input, 'k7.txt' );
reset( input );            { для FreePascal!!! }
readln(s);
```

- при работе в среде FreePascal нужно убедиться, что в параметрах компилятора включена поддержка **длинных символьных строк**; на всякий случай стоит добавить в первой строке программы директиву

```
{$H+}
```

- в языке C++ используем потоки:

```
#include <fstream>
#include <string>
using namespace std;
int main()
{
  ifstream F("k7.txt");
  string s;
  getline( F, s );
  ...
}
```

Самая длинная цепочка символов «С»

- пусть требуется найти самую длинную цепочку символов С (или каких-то других, в соответствии с заданием) в символьной строке **s**;

- можно использовать такой алгоритм:

```
while не конец строки:
  найти очередную букву C
  длина := длина текущей цепочки букв C
  if длина > максимальной длины:
    максимальная длина := длина
```

однако этот алгоритм содержит вложенный цикл и при составлении программы легко запутаться и не учесть какой-то особый случай (например, когда строка состоит только из букв C)

- лучше применить однопроходный алгоритм без вложенного цикла

```
for c in s:
  обработать символ c
```

- будем использовать переменные

cLen – длина текущей цепочки букв C

maxLen – максимальная длина цепочки букв C на данный момент

- рассмотрим очередной символ строки; если это буква C, увеличиваем **cLen** на 1 и, если нужно запоминаем новую максимальную длину; если это не буква C, просто записываем с **cLen** ноль:

```
maxLen = 0
cLen = 0
for c in s:
  if c == 'C':
    cLen += 1           # ещё одна буква C
    if cLen > maxLen:   # возможно, новая максимальная длина
       maxLen = cLen
  else:
    cLen = 0            # цепочка букв C кончилась
```

- проверим правильность работы алгоритма в особых случаях:

а) если вся строка состоит из букв C, значение переменной **cLen** постоянно увеличивается и в конце станет равно длине символьной строки; то же значение окажется и в переменной **maxLen**;

б) если в строке нет символов C, переменная **cLen** всегда равна 0, такое же значение будет и в переменной **maxLen**

Самая длинная цепочка любых символов

- теперь поставим задачу найти самую длинную цепочку символов в символьной строке **s**; сложность состоит в том, что мы (в отличие от предыдущей задачи) не знаем, из каких именно символов состоит самая длинная цепочка
- если символов в алфавите немного (скажем, A, B и C), то можно с помощью описанного выше алгоритма найти самые длинные цепочки из букв A, B и C, а затем выбрать из них «длиннейшую»; такая идея может сработать при аккуратной реализации, но плохо обобщается на случай, когда возможных символов много (например, используются все заглавные латинские буквы и цифры)
- поэтому лучше применить однопроходный алгоритм без вложенного цикла
- будем использовать переменные

curLen – длина текущей цепочки одинаковых символов

maxLen – максимальная длина цепочки одинаковых символов на данный момент

c – символ, из которого строится самая длинная подцепочка

- в начальный момент рассмотрим один первый символ (цепочка длины 1 есть всегда!):

```
maxLen = 1
```

```
curLen = 1
c = s[0]
```

- будем перебирать в цикле все символы, начиная с **s[1]** (второго по счёту) до конца строки, постоянно «оглядываясь назад», на предыдущий символ

```
for i in range(1,len(s)):
  обработать пару символов s[i-1] и s[i]
```

- если очередной символ **s[i]** такой же, как и предыдущий, цепочка одинаковых символов продолжается, и нужно увеличить значение переменной cu**r**Len; если значение cu**r**Len стало больше **maxLen**, обновляем **maxLen** и запоминаем новый базовый символ в переменной **c**:

```
if s[i] == s[i-1]:      # цепочка продолжается
  curLen += 1           # увеличиваем длину
  if curLen > maxLen:   # если цепочка побила рекорд
    maxLen = curLen     # запоминаем её длину...
    c = s[i]            # и образующий символ
else:
  curLen = 1            # началась новая цепочка
```

если очередной символ не совпал с предыдущим, началась новая цепочка, и её длина пока равна 1 (это значение записывается в переменную cu**r**Len)

- получается такой цикл обработки строки:

```
maxLen, curLen, c = 1, 1, s[0]
for i in range(1, len(s)):
  if s[i] == s[i-1]:
    curLen += 1
    if curLen > maxLen:
      maxLen = curLen
      c = s[i]
  else:
    curLen = 1
```

- проверим правильность работы алгоритма в особых случаях:
  - в) если вся строка состоит из одинаковых символов, значение переменной **curLen** постоянно увеличивается и в конце станет равно длине символьной строки; то же значение окажется и в переменной **maxLen**;
  - г) если в строке нет пар одинаковых символов, переменная **curLen** всегда равна 1, такое же значение будет и в переменной **maxLen**

Пример задания:

P-07 (демо-2021). Текстовый файл **24.txt** состоит не более чем из 10^6 символов X, Y и Z. Определите максимальное количество идущих подряд символов, среди которых каждые два соседних различны. Для выполнения этого задания следует написать программу.

Решение:

1) считывание из файла и перебор символов аналогичен задачам P00-P02 (см. ниже).
2) чтобы считать длину цепочки, соответствующей условию, нам нужно будет ввести два счётчика:

   curLen – длина текущей цепочки (которая сейчас обрабатывается)

   maxLen – длина самой длинной на данный момент цепочки в уже обработанной части строки

3) обработка строки сводится к тому, что текущая длина цепочки увеличивается, если соседние символы, **s[i-1]** и **s[i]**, различны; если это не так, сбрасываем длину текущей цепочки в 1
4) можно заметить, что эта задача очень напоминает P-05, только тут обратное условие – нужно искать цепочку, где все соседние символы не одинаковые, а разные, поэтому и решение сводится к изменению условия (см. выделение маркером):

```
with open( "24.txt", "r" ) as F:
  s = F.readline()
maxLen, curLen = 1, 1
for i in range(1, len(s)):
  if s[i] != s[i-1]:
    curLen += 1
    if curLen > maxLen:
      maxLen = curLen
  else:
    curLen = 1
print( maxLen )
```

5) Ответ: 35.
6) программа на **Паскале**:

```
var maxLen, curLen, i: integer;
    s: string;
begin
  assign(input, '24.txt');
  readln(s);
  maxLen := 1;
  curLen := 1;
  for i:=2 to Length(s) do
    if s[i] <> s[i-1] then begin
      curLen := curLen + 1;
      if curLen > maxLen then
        maxLen := curLen;
    end
    else
      curLen := 1;
  writeln(maxLen);
end.
```

7) программа на **C++**:

```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;
int main()
{
  ifstream F("24.txt");
  string s;
  getline( F, s );
  int maxLen = 1, curLen = 1;
  for( int i = 1; i < s.length(); i++ )
    if( s[i] != s[i-1] ) {
      curLen ++;
      if( curLen > maxLen )
```

```
          maxLen = curLen;
        }
      else curLen = 1;
    cout << maxLen;
}
```

Пример задания:

P-06. В текстовом файле **k8.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. **Для каждой цепочки максимальной длины** выведите в отдельной строке сначала символ, из которого строится эта цепочка, а затем через пробел – длину этой цепочки.

Решение:

1) особенность этой задачи в сравнении с P-05 состоит в следующем: если найдено несколько цепочек одинаковой максимальной длины, для каждой из них нужно вывести символ, из которого состоит цепочка, и длину цепочки
2) это значит, что для хранения символа нужна не одна переменная, а массив (в Python – список); если найдена первая цепочка (выполнено условие)
3) итак, теперь **c** – это массив (список); когда найдена первая цепочка максимальной длины (на данный момент), в этот массив записывается символ этой цепочки; если же найдена новая цепочка такой же длины, в массив **добавляется** символ этой цепочки
4) таким образом, в конце прохода в массиве **c** находятся все символы, из которых состоят самые длинные цепочки, и остаётся вывести их на экран; справа от каждого символа выводится длина цепочки:

```
for c1 in c:
  print( c1, maxLen )
```

5) вот полная программа (изменения в сравнении с решением задачи P-05 выделены):

```
with open( "k8.txt", "r" ) as F:
  s = F.readline()
maxLen, curLen, c = 1, 1, [s[0]] # c – массив из одного символа
for i in range(1, len(s)):
  if s[i] == s[i-1]:
    curLen += 1
    if curLen == maxLen: # новая цепочка максимальной длины
      c.append( s[i] )   # добавить символ в массив
    elif curLen > maxLen:
      maxLen = curLen
      c = [s[i]]  # c – массив из одного символа
  else:
    curLen = 1
for c1 in c: # для всех символов в массиве
  print( c1, maxLen ) # вывести символ и длину
```

Решение (программа на языке Pascal):

1) проблема состоит в том, что мы не знаем, сколько цепочек максимальной длины может быть в файле; тут нужен динамический массив (список), для этого далее мы будем использовать язык PascalABC.NET, в котором есть тип данных **List** (список)
2) вначале создаём новый список и записываем у него первый символ строки:

```
var c := new List<char>;
c.Add( s[1] );
```

3) если нашли новую (не первую) цепочку максимальной длины, добавляем новый символ в список:

```
c.Add( s[i] );
```

4) если нашли новую самую длинную цепочку с длиной бОльшей, чем все предыдущие, очищаем список и добавляем в него новый символ.

```
c.Clear;
c.Add( s[i] );
```

5) после окончания обработки нужно вывести все символы и длины цепочек, удобнее всего использовать для этого цикл **foreach**; получается почти так же, как и на Python:

```
foreach var c1 in c do
  writeln( c1, ' ', maxLen );
```

6) вот полная программа:

```
var maxLen, curLen, i: integer;
    s: string;
begin
  assign(input, 'k8.txt');
  readln(s);
  maxLen := 1;
  curLen := 1;
  var c := new List<char>;
  c.Add( s[1] );
  for i:=2 to Length(s) do
    if s[i] = s[i-1] then begin
      curLen := curLen + 1;
      if curLen = maxLen then begin
        c.Add( s[i] );
      end
      else if curLen > maxLen then begin
        maxLen := curLen;
        c.Clear;
        c.Add( s[i] );
      end
    end
    else
      curLen := 1;
  foreach var c1 in c do
    writeln( c1, ' ', maxLen );
end.
```

Решение (программа на языке C++):

1) динамический массив (список), для этого далее мы будем использовать тип данных **list** (список) из библиотеки STL; не забудьте, что нужно подключить заголовочный файл list:

```
#include <list>
```

2) вначале создаём список символов, состоящий из одного первого символа строки:

```
list<char> c( s[0] );
```

3) если нашли новую (не первую) цепочку максимальной длины, добавляем новый символ в список с помощью метода **push_back**:

```
c.push_back( s[i] );
```

4) если нашли новую самую длинную цепочку с длиной бОльшей, чем все предыдущие, очищаем список и добавляем в него новый символ:

```
c.clear();
c.push_back( s[i] );
```

5) после окончания обработки нужно вывести все символы и длины цепочек, удобнее всего использовать для этого особую форму цикла **for**; получается почти так же, как и на Python:

```
for( char c1:  c )
  cout << c1 << ' ' << maxLen << endl;
```

6) вот полная программа:

```
#include <iostream>
#include <fstream>
#include <string>
#include <list>
using namespace std;
int main()
{
  ifstream F("k8.txt");
  string s;
  getline( F, s );
  int maxLen = 1, curLen = 1;
  list<char> c(s[0]);
  for( int i = 1; i < s.length(); i++ )
    if( s[i] == s[i-1] ) {
      curLen ++;
    if( curLen == maxLen )
      c.push_back(s[i]);
    else if( curLen > maxLen ) {
      maxLen = curLen;
      c.clear();
      c.push_back( s[i] );
      }
    }
   else curLen = 1;
for( char c1:  c )
  cout << c1 << ' ' << maxLen << endl;
}
```

Ещё пример задания:

P-05. В текстовом файле **k8.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Выведите сначала символ, из которого строится цепочка, а затем через пробел – длину этой цепочки.

Замечание (Б.С. Михлин). Может случиться так, что в файле будут несколько самых длинных цепочек (одинаковой длины), состоящих из разных символов. На этот случай условие задачи требует уточнения – какой именно символ выводить в ответе? Далее мы будем считать, что в этом случае нужно вывести символ, который формирует первую цепочку максимальной длины.

Решение:

1) сначала нужно открыть файл и прочитать все символы в символьную строку:

```
with open("k8.txt", "r") as F:
  s = F.readline()
```

2) теперь задача свелась к определению наибольшего количества подряд идущих одинаковых символов в символьной строке **s** (этот алгоритм приведён в начале файла)
3) полная программа на языке Python:

```
with open( "k8.txt", "r" ) as F:
  s = F.readline()
maxLen, curLen, c = 1, 1, s[0]
for i in range(1, len(s)):
  if s[i] == s[i-1]:
    curLen += 1
    if curLen > maxLen:
      maxLen = curLen
      c = s[i]
  else:
    curLen = 1
print( c, maxLen )
```

Обратим внимание, что условие

```
if curLen > maxLen: ...
```

гарантирует, что будет запомнена именно первая цепочка максимальной длины, так как это условие выполнится, когда новая цепочка строго длиннее предыдущего «рекорда». Если бы нужно было вывести символ, формирующий **последнюю** из самых длинных цепочек, неравенство следовало бы сделать нестрогим:

```
if curLen >= maxLen: ...
```

Решение (программа на языке Pascal):

7) далее мы будем использовать язык PascalABC.NET, который обладает двумя важными достоинствами, упрощающими решение таких задач:
   - позволяет легко переключать входной поток с консоли на нужный файл:

```
assign( input, 'k8.txt' );
```

     теперь можно писать программу так же, как и при вводе данных с клавиатуры, а она на самом деле будет читать их из указанного файла;
   - не имеет ограничения на длину строк (переменных типа **string**); в устаревших версиях языка Pascal длина строки не может превышать 255 символов
8) читаем одну строку из файла в строковую переменную **s**:

```
readln(s);
```

9) теперь можно в цикле перебрать все символы строки **s**, начиная **со второго** (чтобы сравнивать его с предыдущим):

```
for i:=2 to Length(s) do
  обработать пару s[i-1] и s[i]
```

10) обработка выполняется по алгоритму, описанному выше (см. программу на Python)
11) полная программа на языке Pascal:

```
var maxLen, curLen, i: integer;
    s: string;
    c: char;
begin
  assign(input, 'k8.txt');
  readln(s);
  maxLen := 1;
  curLen := 1;
```

```
  c := s[1];
  for i:=2 to Length(s) do
    if s[i] = s[i-1] then begin
      curLen := curLen + 1;
      if curLen > maxLen then begin
        maxLen := curLen;
        c := s[i];
      end
    end
    else
      curLen := 1;
  writeln(c, ' ', maxLen);
end.
```

Решение (программа на языке C++):

1) в C++ удобно работать с файлами через файловые потоки; для того, чтобы использовать эту возможность, нужно подключить заголовочный файл **fstream**:

```
#include <fstream>
```

2) теперь можно открыть файловый поток, связать его с нужным файлом и прочитать из потока строку в переменную типа **string**:

```
ifstream F("k8.txt");
string s;
getline( F, s );
```

3) алгоритм обработки строки тот же, что и в программах на языках Python и Pascal, рассмотренных выше

4) поскольку в программе используется много объектов из пространства имён **std**, имеет смысл подключить его в начале программы:

```
using namespace std;
```

5) полная программа на языке C++:

```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;
int main()
{
  ifstream F("k8.txt");
  string s;
  getline( F, s );
  int maxLen = 1, curLen = 1;
  char c = s[0];
  for( int i = 1; i < s.length(); i++ )
    if( s[i] == s[i-1] ) {
      curLen ++;
      if( curLen > maxLen ) {
        maxLen = curLen;
        c = s[i];
        }
      }
    else curLen = 1;
  cout << c << ' ' << maxLen;
}
```

Ещё пример задания:

P-04. (А.М. Кабанов) В текстовом файле k7.txt находится цепочка из символов латинского алфавита A, B, C, D, E. Найдите количество цепочек длины 3, удовлетворяющих следующим условиям:

- 1-й символ – один из символов B, C или D;
- 2-й символ – один из символов B, D, E, который не совпадает с первым;
- 3-й символ – один из символов B, C, E, который не совпадает со вторым.

Решение:

6) Считывание из файла и перебор символов аналогичен задачам P00-P02 (см. ниже).
7) Переберём все тройки символов. Примем, что переменная i будет хранить номер первого элемента в тройке, то есть, будем рассматривать тройки (`s[i]`, `s[i+1]`, `s[i+2]`).
8) Организуем цикл который перебирает значения `i` от 1 до `len(s)-2`

```
for i in range(len(s)-2):
  ...
```

9) Проверяем символы в каждой тройке на соответствие условию. Проверка принадлежности символов набору аналогична заданию 1. Дополнительно необходимо указать условия неравенства символов, указанных в условии задачи. Если условия выполняются, то к переменной количества прибавляется единица.
10) полная программа на Python:

```
s = open('k7.txt').read()
count = 0
for i in range(len(s)-2):
    if s[i] in 'BCD' and s[i+1] in 'BDE' \
      and s[i+2] in 'BCE' and s[i]!=s[i+1] \
      and s[i+1]!=s[i+2]:
      count += 1
print(count)
```

Решение (программа на языке PascalABC.NET):

```
begin
  var s: string;
  var i, count: integer;
  assign(input,'k7.txt');
  readln(s);
  count:=0;
  for i:=1 to Length(s)-2 do
    if (s[i] in 'BCD') and (s[i+1] in 'BDE')
       and (s[i+2] in 'BCE') and (s[i]<>s[i+1])
       and (s[i+1]<>s[i+2]) then
      count := count+1;
  writeln(count);
end.
```

Решение (программа на языке C++):

```
using namespace std;
#include <iostream>
#include <fstream>
#include <string>
int main(){
```

```
  ifstream F("k7.txt");
  string s;
  getline( F, s );
  int count = 0;
  for( int i = 0; i < s.length()-2; i++ )
    if( (s[i]=='B' || s[i]=='C' || s[i]=='D')
       && (s[i+1]=='B' || s[i+1]=='D' || s[i+1]=='E')
       && (s[i+2]=='B' || s[i+2]=='C' || s[i+2]=='E')
       && s[i]!=s[i+1] && s[i+1]!=s[i+2] )
        count++;
  cout << count;
}
```

Ещё пример задания:

P-03. (А.М. Кабанов) В текстовом файле **k7.txt** находится цепочка из символов латинского алфавита A, B, C, D, E. Найдите максимальную длину цепочки вида EABEABEABE... (составленной из фрагментов EAB, **последний фрагмент может быть неполным**).

Решение:

1) Считывание из файла и перебор символов аналогичен задачам P00-P02 (см. ниже).
2) Проверка того, что символ принадлежит цепочке, производится следующим образом. Заметим, что в искомой цепочке чередуется группа из трёх символов (**EAB**). Пронумеруем символы искомой цепочки, начиная с нуля.

Символ	E	A	B	E	A	B	E	A
Count	0	1	2	3	4	5	6	7
Count%3	0	1	2	0	1	2	0	1

3) Видно, что позиция каждого символа имеет одинаковый остаток от деления на 3. Позиция есть значения переменной счётчика в момент проверки символа. Поэтому если совпадает символ и соответствующий ему остаток от деления, то он принадлежит цепочке. Для приведённого примера условие проверки выглядит так

```
if (char == 'E' and count%3==0) or \
   (char == 'A' and count%3==1) or \
   (char == 'B' and count%3==2):
```

4) Если символ не является частью этой цепочки, но может являться её началом (E), длина цепочки принимается равной единице, в противном случае длина обнуляется

```
elif (char=='E'):
    count = 1
else:
    count = 0
```

5) Полная программа на языке Python:

```
s = open('k7.txt').read()
count = 0
maxCount = 0
for char in s:
    if (char == 'E' and count%3 == 0) or \
       (char == 'A' and count%3 == 1) or \
       (char == 'B' and count%3 == 2):
        count += 1
        if count > maxCount:
            maxCount = count
```

```
    elif (char=='E'):
        count = 1
    else:
        count = 0
print(maxCount)
```

Решение (полная программа на языке PascalABC.NET):

```
begin
  var s: string;
  var i, count, maxCount: integer;
  assign(input,'k7-2.txt');
  readln(s);
  count:=0;
  maxCount:=0;
  for i:=1 to Length(s) do
    if ((s[i]='E') and (count mod 3=0)) or
       ((s[i]='A') and (count mod 3=1)) or
       ((s[i]='B') and (count mod 3=2)) then  begin
      count := count+1;
      if count > maxCount then
        maxCount := count;
    end
    else if s[i]='E' then count:=1
    else count := 0;

    writeln(maxCount);
end.
```

Решение (полная программа на языке C++):

```
using namespace std;
#include <iostream>
#include <fstream>
#include <string>
int main(){
  ifstream F("k7-2.txt");
  string s;
  getline( F, s );
  int count = 0, maxCount = 0;
  for( int i = 0; i < s.length(); i++ )
    if( (s[i] == 'E' && count%3 == 0) ||
        (s[i] == 'A' && count%3 == 1) ||
        (s[i] == 'B' && count%3 == 2) ) {
      count ++;
      if( count > maxCount )
        maxCount = count;
      }
    else if(s[i]== 'E') count = 1
    else count = 0
  cout << maxCount;
}
```

Ещё пример задания:

P-02. (А.М. Кабанов) В текстовом файле **k7.txt** находится цепочка из символов латинского алфавита A, B, C, D, E. Найдите длину самой длинной подцепочки, состоящей из символов A, B или C (в произвольном порядке).

Решение:

1) сначала нужно открыть файл и прочитать все символы в символьную строку:
```
s = open('k7.txt').read()
```
2) теперь можно в цикле перебрать все символы строки s:
```
for char in s:
  …
```
3) теперь задача свелась к определению наибольшей подстроки, состоящей из символов A, B или C, в символьной строке **s**.
4) Проверку того, что символ – один из набора A, B, C удобно записывать с помощью условия
```
if char in 'ABC':
```
5) Полная программа на языке Python:
```
s = open('k7.txt').read()
count = 0
maxCount = 0
for char in s:
  if char in 'ABC':
    count += 1
    if count>maxCount:
      maxCount = count
    else:
      count=0
print(maxCount)
```

Решение (программа на языке PascalABC.NET):

1) В начале переключим входной поток с консоли на нужный файл, а затем считаем одну строку из файла в строковую переменную **s**
```
assign(input,'k7.txt');
readln(s);
```
2) теперь можно в цикле перебрать все символы строки s:
```
for i:=1 to Length(s) do
  …
```
3) обработка символа выполняется по алгоритму, описанному выше (см. программу на Python)
4) Проверка того, что символ – один из набора A, B, C – в PascalABC.NET записывается аналогично
```
if s[i] in 'ABC' then
```
а в среде FreePascal придётся использовать старинный способ:
```
if s[i] in ['A','B','C'] then
```
5) Полная программа на языке PascalABC.NET:
```
begin
  var s: string;
  var i, count, maxCount: integer;
  assign(input,'k7.txt');
  readln(s);
  count:=0;
  maxCount:=0;
```

```
  for i:=1 to Length(s) do
    if s[i] in 'ABC' then  { if s[i] in ['A','B','C'] }
    begin
      count := count+1;
      if count > maxCount then
        maxCount := count;
    end
    else
      count := 0;
    writeln(maxCount);
end.
```

Решение (программа на языке C++):

1) Для чтения из файла подключим заголовочный файл **fstream**, откроем файловый поток и считаем его в строковую переменную **s**

```
#include <fstream>
#include <string>
…
ifstream F("k7.txt");
string s;
getline( F, s );
```

2) алгоритм обработки строки тот же, что и в программах на языках Python и Pascal, рассмотренных выше, однако проверка того, что символ – один из набора A, B, C записывается по-другому

```
if( s[i]=='A' || s[i]=='B' || s[i]=='C' )
```

3) Полная программа на языке C++:

```
using namespace std;
#include <iostream>
#include <fstream>
#include <string>
int main(){
  ifstream F("k7.txt");
  string s;
  getline( F, s );
  int count = 0, maxCount = 0;
  for( int i = 0; i < s.length(); i++ )
    if( s[i] == 'A' || s[i] == 'B' || s[i] == 'C' ) {
      count ++;
   if( count > maxCount )
        maxCount = count;
 }
 else count = 0;
  cout << maxCount;
}
```

Ещё пример задания:

P-01. В текстовом файле **k7.txt** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.

Решение:

1) сначала нужно открыть файл и прочитать все символы в символьную строку:

```
with open("k7.txt", "r") as F:
  s = F.readline()
```

2) теперь задача свелась к определению наибольшего количества подряд идущих букв C в символьной строке **s** (этот алгоритм приведён в начале файла)
3) полная программа на языке Python:

```
with open( "k7.txt", "r" ) as F:
  s = F.readline()
maxLen, cLen = 0, 0
for c in s:
  if c == 'C':
    cLen += 1
    if cLen > maxLen:
       maxLen = cLen
  else:
    cLen = 0
print( maxLen )
```

Решение (программа на языке Pascal):

7) далее мы будем использовать язык PascalABC.NET, который обладает двумя важными достоинствами, упрощающими решение таких задач:
   - позволяет легко переключать входной поток с консоли на нужный файл:

     ```
     assign( input, 'k7.txt' );
     ```

     теперь можно писать программу так же, как и при вводе данных с клавиатуры, а она на самом деле будет читать их из указанного файла;
   - не имеет ограничения на длину строк (переменных типа **string**); в устаревших версиях языка Pascal длина строки не может превышать 255 символов
8) читаем одну строку из файла в строковую переменную **s**:

```
readln(s);
```

9) теперь можно в цикле перебрать все символы строки **s**:

```
for i:=1 to Length(s) do
  обработать s[i]
```

10) обработка символа выполняется по алгоритму, описанному выше (см. программу на Python)
11) полная программа на языке Pascal:

```
var maxLen, cLen, i: integer;
    s: string;
begin
  assign(input, 'k7.txt');
  readln(s);
  maxLen := 0;
  cLen := 0;
  for i:=1 to Length(s) do
    if s[i] = 'C' then begin
      cLen := cLen + 1;
      if cLen > maxLen then maxLen := cLen;
    end
    else
      cLen := 0;
  writeln(maxLen);
end.
```

Решение (программа на языке C++):

6) в C++ удобно работать с файлами через файловые потоки; для того, чтобы использовать эту возможность, нужно подключить заголовочный файл **fstream**:
```
#include <fstream>
```
7) для того чтобы читать всю строку целиком с помощью функции **getline**, нужно подключить заголовочный файл **string**:
```
#include <string>
```
8) теперь можно открыть файловый поток, связать его с нужным файлом и прочитать из потока строку в переменную типа **string**:
```
ifstream F("k7.txt");
string s;
getline( F, s );
```
9) алгоритм обработки строки тот же, что и в программах на языках Python и Pascal, рассмотренных выше
10) поскольку в программе используется много объектов из пространства имён **std**, имеет смысл подключить его в начале программы:
```
using namespace std;
```
11) полная программа на языке C++:
```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;
int main()
{
  ifstream F("k7.txt");
  string s;
  getline( F, s );
  int maxLen = 0, cLen = 0;
  for( int i = 0; i < s.length(); i++ )
    if( s[i] == 'C' ) {
      cLen ++;
      if( cLen > maxLen )
        maxLen = cLen;
      }
    else cLen = 0;
  cout << maxLen;
}
```

Решение методом грубой силы (Б.С. Михлин):

1) если решить красиво не получается, можно применить метод грубой силы, использующий встроенную функцию поиска подстроки: ищём строку из одного символа C, потом – из двух символов, из трёх и т.д.; в какой-то момент поиск не даст результата, значит ответ – это длина предыдущей цепочки, которая короче текущей на единицу
2) вот решение на Python:
```
with open( "k7.txt", "r" ) as F:
  s = F.readline()
cc = 'C'
while cc in s:
  cc += 'C'
print( len(cc)-1 )
```
3) решение на Паскале:

```
var cc, s: string;
begin
  assign(input, 'k7.txt');
  readln(s);
  cc := 'C';
  while Pos(cc, s) > 0 do
    cc := cc + 'C';
  writeln( Length(cc)-1 );
end.
```

4) решение на C++:

```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;
int main()
{
  ifstream F("k7.txt");
  string s, cc;
  getline( F, s );
  cc = 'C';
  while( s.find(cc) != string::npos )
    cc += 'C';
  cout << cc.length()-1;
}
```

5) эту задачу можно решать вообще без программирования, используя функцию поиска в любом текстовом редакторе или процессоре; для ускорения можно сначала удваивать длину искомой цепочки, а после того, как поиск закончится неудачно, применять двоичный поиск в интервале
6) конечно, нужно понимать, что эффективность (скорость работы) этого алгоритма крайне низкая в сравнении с описанным выше однопроходным поиском, но на небольших файлах и этот метод вполне может сработать.

Решение в электронных таблицах (Б.С. Михлин):

1) можно применить тот же метод грубой силы, использующий электронные таблицы. Сначала откроем файл в текстовом редакторе и скопируем все его содержимое в буфер обмена. Затем откроем новую электронную таблицу и вставим строку из буфера обмена в какую-нибудь ячейку (в примере ниже это ячейка A2). Затем в окне «Найти» вбиваем один символ «C» и нажимаем кнопку «Найти все», потом два символа «C», три и т.д., пока не появится сообщение «…не удается найти искомые данные». Значит максимальная длина подцепочки из символов «C», входящая в заданную цепочку, на единицу меньше. При большой длине максимальной подцепочки при подсчете в ней количества символов легко ошибиться.
2) можно также использовать встроенные текстовые функции электронных таблиц: FIND (НАЙТИ) или SEARCH (ПОИСК) и REPT (ПОВТОР). Меняя в функции ПОВТОР коэффициент повторения символа "C" мы повторяем идею п. 1. Для ускорения поиска можно коэффициент повторения менять сперва с крупным шагом, а затем с более мелким. Также можно обойтись только одной ячейкой с формулой.
   Функции НАЙТИ и ПОИСК выводят положение начала искомой подцепочки в заданной цепочке символов или сообщение #ЗНАЧ!, если подцепочка не найдена. Если поиск надо осуществлять с начала цепочки, то третий параметр функций НАЙТИ и ПОИСК можно не

указывать. Функция НАЙТИ учитывает регистр символов. Функция ПОИСК не учитывает регистр символов и в ней можно использовать подстановочные символы (* и ?).

Вот решение задачи в OpenOffice Calc:

	A	B	C
1			
2	CCCBACCBCCCCCACCABCACCACCCCAAABAABBBCCCCBCCCCBBBCBCBACCACCABCACCBBBBCACBCCCCCACCCCCBAACAC		
3	**Результат формулы**	**Формула**	**Комментарии** В ячейке A2 находится заданная цепочка из файла K7-53.txt. (Ее длина не должна превышать 32767)
4		**FIND**	
5	1	FIND(REPT("C";1);A2)	Подцепочка из одного символа "C" начинается с позиции 1
6	354	FIND(REPT("C";10);A2)	Подцепочка из десяти символов "C" начинается с позиции 354
7	#ЗНАЧЕН!	FIND(REPT("C";20);A2)	Подцепочка из двадцати символов "C" не найдена.
8	#ЗНАЧЕН!	FIND(REPT("C";15);A2)	Подцепочка из пятнадцати символов "C" тоже не найдена.
9	763	FIND(REPT("C";14);A2)	Подцепочка из четырнадцати символов "C" начинается с позиции 763. Т.е. она входит в заданную цепочку.
10	**Ответ:**	**14**	Максимальная длина подцепочки из символов "C"
11		**SEARCH**	
12	1	SEARCH(REPT("C";1);A2)	Подцепочка из одного символа "C" начинается с позиции 1
13	354	SEARCH(REPT("C";10);A2)	Подцепочка из десяти символов "C" начинается с позиции 354
14	#ЗНАЧЕН!	SEARCH(REPT("C";20);A2)	Подцепочка из двадцати символов "C" не найдена.
15	#ЗНАЧЕН!	SEARCH(REPT("C";15);A2)	Подцепочка из пятнадцати символов "C" тоже не найдена.
16	763	SEARCH(REPT("C";14);A2)	Подцепочка из четырнадцати символов "C" начинается с позиции 763. Т.е. она входит в заданную цепочку.
17	**Ответ:**	**14**	Максимальная длина подцепочки из символов "C"

и в русской версии Excel:

	A	B	C
1			
2	CCCBACCBCCCCCACCABCACCACCCCAAABAABBBCCCCBCCCCBBBCBCBACCACCABCACCBBBBCACBCCCCCACCCCCBAAC		
3	**Результат формулы**	**Формула**	**Комментарии** В ячейке A2 находится заданная цепочка из файла K7-53.txt. (Ее длина не должна превышать 32767)
4		**НАЙТИ**	
5	1	НАЙТИ(ПОВТОР("C";1);A2)	Подцепочка из одного символа "C" начинается с позиции 1
6	354	НАЙТИ(ПОВТОР("C";10);A2)	Подцепочка из десяти символов "C" начинается с позиции 354
7	#ЗНАЧ!	НАЙТИ(ПОВТОР("C";20);A2)	Подцепочка из двадцати символов "C" не найдена.
8	#ЗНАЧ!	НАЙТИ(ПОВТОР("C";15);A2)	Подцепочка из пятнадцати символов "C" тоже не найдена.
9	763	НАЙТИ(ПОВТОР("C";14);A2)	Подцепочка из четырнадцати символов "C" начинается с позиции 763. Т.е. она входит в заданную цепочку.
10	**Ответ:**	**14**	Максимальная длина подцепочки из символов "C"
11		**ПОИСК**	
12	1	ПОИСК(ПОВТОР("C";1);A2)	Подцепочка из одного символа "C" начинается с позиции 1
13	354	ПОИСК(ПОВТОР("C";10);A2)	Подцепочка из десяти символов "C" начинается с позиции 354
14	#ЗНАЧ!	ПОИСК(ПОВТОР("C";20);A2)	Подцепочка из двадцати символов "C" не найдена.
15	#ЗНАЧ!	ПОИСК(ПОВТОР("C";15);A2)	Подцепочка из пятнадцати символов "C" тоже не найдена.
16	763	ПОИСК(ПОВТОР("C";14);A2)	Подцепочка из четырнадцати символов "C" начинается с позиции 763. Т.е. она входит в заданную цепочку.
17	**Ответ:**	**14**	Максимальная длина подцепочки из символов "C"

Задачи для тренировки:

1) В текстовом файле[1] **`k7-0.txt`** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
2) В текстовом файле **`k7-3.txt`** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
3) В текстовом файле **`k7-5.txt`** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
4) В текстовом файле **`k7-20.txt`** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.

[1] Архив с файлами данных для этой и следующих задач можно скачать по ссылке http://kpolyakov.spb.ru/download/24data.zip.

5) В текстовом файле **k7-25.txt** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
6) В текстовом файле **k7-29.txt** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
7) В текстовом файле **k7-40.txt** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
8) В текстовом файле **k7-42.txt** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
9) В текстовом файле **k7-44.txt** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
10) В текстовом файле **k7-45.txt** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
11) В текстовом файле **k7-53.txt** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
12) В текстовом файле **k7-70.txt** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
13) В текстовом файле **k7-75.txt** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
14) В текстовом файле **k7-80.txt** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
15) В текстовом файле **k7-84.txt** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
16) В текстовом файле **k7-91.txt** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
17) В текстовом файле **k7-94.txt** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
18) В текстовом файле **k7-96.txt** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
19) В текстовом файле **k7-97.txt** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
20) В текстовом файле **k7-100.txt** находится цепочка из символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C.
21) (**А.М. Кабанов**) В текстовом файле **k7a-1.txt** находится цепочка из символов латинского алфавита A, B, C, D, E. Найдите длину самой длинной подцепочки, состоящей из символов A, B или C (в произвольном порядке).
22) (**А.М. Кабанов**) В текстовом файле **k7a-2.txt** находится цепочка из символов латинского алфавита A, B, C, D, E, F. Найдите длину самой длинной подцепочки, состоящей из символов A, C, D (в произвольном порядке).
23) (**А.М. Кабанов**) В текстовом файле **k7a-3.txt** находится цепочка из символов латинского алфавита A, B, C, D, E, F. Найдите длину самой длинной подцепочки, состоящей из символов A, B, E, F (в произвольном порядке).
24) (**А.М. Кабанов**) В текстовом файле **k7a-4.txt** находится цепочка из символов латинского алфавита A, B, C, D, E, F. Найдите длину самой длинной подцепочки, не содержащей символа D.
25) (**А.М. Кабанов**) В текстовом файле **k7a-5.txt** находится цепочка из символов латинского алфавита A, B, C, D, E, F. Найдите длину самой длинной подцепочки, не содержащей символов C и F.

26) (**А.М. Кабанов**) В текстовом файле **k7a-6.txt** находится цепочка из символов латинского алфавита A, B, C, D, E, F. Найдите длину самой длинной подцепочки, не содержащей гласных букв.
27) (**А.М. Кабанов**) В текстовом файле **k7b-1.txt** находится цепочка из символов латинского алфавита A, B, C, D, E. Найдите максимальную длину цепочки вида EABEABEABE... (состоящей из фрагментов EAB, последний фрагмент может быть неполным).
28) (**А.М. Кабанов**) В текстовом файле **k7b-2.txt** находится цепочка из символов латинского алфавита A, B, C, D, E, F. Найдите максимальную длину цепочки вида DBACDBACDBAC....
29) (**А.М. Кабанов**) В текстовом файле **k7b-3.txt** находится цепочка из символов латинского алфавита A, B, C, D, E, F. Найдите максимальную длину цепочки вида BAFEBAFEBAFE...
30) (**А.М. Кабанов**) В текстовом файле **k7b-4.txt** находится цепочка из символов латинского алфавита A, B, C, D, E, F. Найдите максимальную длину цепочки вида EBCFEBCFEBCF....
31) (**А.М. Кабанов**) В текстовом файле **k7b-5.txt** находится цепочка из символов латинского алфавита A, B, C, D, E, F. Найдите максимальную длину цепочки вида CACACA....
32) (**А.М. Кабанов**) В текстовом файле **k7b-6.txt** находится цепочка из символов латинского алфавита A, B, C, D, E, F. Найдите максимальную длину цепочки вида DAFDAFDAF....
33) (**А.М. Кабанов**) В текстовом файле **k7c-1.txt** находится цепочка из символов латинского алфавита A, B, C, D, E. Найдите количество цепочек длины 3, удовлетворяющих следующим условиям:
    - 1-й символ – один из символов B, C или D;
    - 2-й символ – один из символов B, D, E, который не совпадает с первым;
    - 3-й символ – один из символов B, C, E, который не совпадает со вторым.
34) (**А.М. Кабанов**) В текстовом файле **k7c-2.txt** находится цепочка из символов латинского алфавита A, B, C, D, E, F. Найдите количество цепочек длины 3, удовлетворяющих следующим условиям:
    - 1-й символ – один из A, C, E;
    - 2-й символ – один из A, D, F, который не совпадает с первым;
    - 3-й символ – один из A, B, F, который не совпадает со вторым.
35) (**А.М. Кабанов**) В текстовом файле **k7c-3.txt** находится цепочка из символов латинского алфавита A, B, C, D, E, F. Найдите количество цепочек длины 3, удовлетворяющих следующим условиям:
    - 2-й символ – один из B, D, E;
    - 3-й символ – один из A, C, D, который не совпадает со вторым;
    - 1-й символ – совпадает с третьим.
36) (**А.М. Кабанов**) В текстовом файле **k7c-4.txt** находится цепочка из символов латинского алфавита A, B, C, D, E, F. Найдите количество цепочек длины 3, удовлетворяющих следующим условиям:
    - 3-й символ – один из C, D, F;
    - 1-й символ – один из A, D, F, который не совпадает с третьим;
    - 2-й символ – один из C, D, F, который не совпадает с третьим.
37) (**А.М. Кабанов**) В текстовом файле **k7c-5.txt** находится цепочка из символов латинского алфавита A, B, C, D, E, F. Найдите количество цепочек длины 5, в которых соседние символы не совпадают.
38) (**А.М. Кабанов**) В текстовом файле **k7c-6.txt** находится цепочка из символов латинского алфавита A, B, C, D, E, F. Найдите количество цепочек длины 3, в которых символы не совпадают.
39) (**Б.С. Михлин**) В текстовом файле **k7-m1.txt** находится цепочка из прописных (заглавных) символов латинского алфавита A, B, C. Найдите длину самой короткой подцепочки, состоящей из

символов C (C-подцепочки). В ответе через пробел укажите: длину найденной подцепочки (если C-подцепочек нет, то 0), количество C-подцепочек и длину исходной цепочки.

40) (**Б.С. Михлин**) В текстовом файле **k7-m2.txt** находится цепочка из прописных (заглавных) символов латинского алфавита A, B, C. Найдите длину самой длинной подцепочки, состоящей из символов C (C-подцепочки). В ответе через пробел укажите: длину найденной подцепочки (если C-подцепочек нет, то 0), количество C-подцепочек и длину исходной цепочки.
41) (**Б.С. Михлин**) В текстовом файле **k7-m3.txt** находится цепочка из прописных (заглавных) символов латинского алфавита A, B, C. Найдите все подцепочки, состоящие из символов C (C-подцепочки) длиной не более четырех. В ответе через пробел укажите: порядковый номер найденной подцепочки (начиная с единицы) при проходе по исходной цепочке слева направо, длину подцепочки и саму подцепочку, заменив в ней, начиная со второго символа «C», большие «C» на «c» строчные (маленькие). Гарантируется, что в исходной цепочке есть C-подцепочки.
42) (**Б.С. Михлин**) В текстовом файле **k7-m4.txt** находится цепочка из прописных (заглавных) символов латинского алфавита A, B, C. Найдите все подцепочки, состоящие из символов C (C-подцепочки) длиной не менее шести. В ответе через пробел укажите: порядковый номер найденной подцепочки (начиная с единицы) при проходе по исходной цепочке СПРАВА НАЛЕВО, ее длину и саму подцепочку, заменив в ней все символы «C» слева от правого символа «C» на «c» строчное (маленькое). Гарантируется, что в исходной цепочке есть C-подцепочки.
43) (**Б.С. Михлин**) В текстовом файле **k7-m5.txt** находится цепочка из прописных (заглавных) символов латинского алфавита A, B, C. В исходной цепочке замените все найденные C-подцепочки на подцепочки, содержащие длину текущей C-подцепочки с последующей текущей C-подцепочкой с замененными символами «C» большими на «c» маленькие. В ответе в трех строчках выведите:
    1) количество C-подцепочек;
    2) левые 15 символов, пробел и правые 15 символов исходной цепочки;
    3) левые 15 символов, пробел и правые 15 символов преобразованной цепочки.
44) (**Б.С. Михлин**) В текстовом файле **k7-m6.txt** находится цепочка из прописных (заглавных) символов латинского алфавита A, B, C. В исходной цепочке замените все найденные C-подцепочки на подцепочки, содержащие порядковый номер (начиная с единицы) текущей C-подцепочки с последующей текущей C-подцепочкой в которой символы «C», начиная со второго, заменены на восклицательные знаки («!»). В ответе в трех строчках выведите:
    1) количество C-подцепочек;
    2) левые 15 символов, пробел и правые 15 символов исходной цепочки;
    3) левые 15 символов, пробел и правые 15 символов преобразованной цепочки.
45) (**Б.С. Михлин**) В текстовом файле **k7-m7.txt** находится цепочка из прописных (заглавных) символов латинского алфавита A, B, C. В исходной цепочке все найденные C-подцепочки переместите в начало исходной цепочки и перед ними поставьте суммарную длину C-подцепочек, а после произведение длин C-подцепочек. Гарантируется, что в исходной цепочке есть C-подцепочки. В ответе в трех строчках выведите:
    1) количество C-подцепочек;
    2) левые 35 символов исходной цепочки;
    3) левые 35 символов преобразованной цепочки.
46) (**Б.С. Михлин**) В текстовом файле **k7-m21.txt** находится цепочка из прописных (заглавных) символов латинского алфавита A, B, C, D, E, F. Найдите количество подцепочек из трех символов, в которых символы идут в алфавитном порядке и номер начала последней найденной подцепочки (первый символ исходной цепочки имеет номер 0).

Например, у цепочки ABCDF таких подцепочек три: ABC, BCD и CDF, а номер начала последней найденной подцепочки (CDF) два и, следовательно, ответ: 3 2.

47) (**Б.С. Михлин**) В текстовом файле **k7-m22.txt** находится цепочка из прописных (заглавных) символов латинского алфавита A, B, C, D, E, F. Найдите количество подцепочек из трех символов, в которых символы идут в обратном алфавитном порядке и номер начала последней найденной подцепочки (первый символ исходной цепочки имеет номер 0).
Например, у цепочки FDCBA таких подцепочек три: FDC, DCB и CBA, а номер начала последней найденной подцепочки (CBA) два и, следовательно, ответ: 3 2.
48) (**Б.С. Михлин**) В текстовом файле **k7-m23.txt** находится цепочка из прописных (заглавных) символов латинского алфавита A, B, C, D, E, F. Найдите количество подцепочек из трех символов, в которых символы идут не в убывающем алфавитном порядке и номер начала последней найденной подцепочки (первый символ исходной цепочки имеет номер 0).
Например, у цепочки ABCFF таких подцепочек три: ABC, BCF и CFF, а номер начала последней найденной подцепочки (CFF) два и, следовательно, ответ: 3 2.
49) (**Б.С. Михлин**) В текстовом файле **k7-m24.txt** находится цепочка из прописных (заглавных) символов латинского алфавита A, B, C, D, E, F. Найдите количество подцепочек из трех символов, в которых символы идут не в возрастающем алфавитном порядке и номер начала последней найденной подцепочки (первый символ исходной цепочки имеет номер 0).
Например, у цепочки FFCBA таких подцепочек три: FFC, FCB и CBA, а номер начала последней найденной подцепочки (CBA) два и, следовательно, ответ: 3 2.
50) (**Б.С. Михлин**) В текстовом файле **k7-m25.txt** находится цепочка из прописных (заглавных) символов латинского алфавита A, B, C, D, E, F. Найдите количество подцепочек из трех символов, в которых средний символ ближе к концу алфавита, чем символ слева и справа от него, а также найдите номер начала последней найденной подцепочки (первый символ исходной цепочки имеет номер 0). Например, у цепочки ACBFAED таких подцепочек три: ACB, BFA и AED, а номер начала последней найденной подцепочки (AED) четыре и, следовательно, ответ: 3 4.
51) (**Б.С. Михлин**) В текстовом файле **k7-m26.txt** находится цепочка из прописных (заглавных) символов латинского алфавита A, B, C, D, E, F. Найдите количество подцепочек из трех символов, в которых средний символ ближе к началу алфавита, чем символ слева и справа от него, а также найдите номер начала последней найденной подцепочки (первый символ исходной цепочки имеет номер 0).
Например, у цепочки FABACAE таких подцепочек три: FAB, BAC и CAE, а номер начала последней найденной подцепочки (CAE) четыре и, следовательно, ответ: 3 4.
52) В текстовом файле **k8-0.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файл несколько цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.
53) В текстовом файле **k8-4.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файл несколько цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.
54) В текстовом файле **k8-6.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек

одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.

55) В текстовом файле **k8-12.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.
56) В текстовом файле **k8-18.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.
57) В текстовом файле **k8-20.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.
58) В текстовом файле **k8-25.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.
59) В текстовом файле **k8-31.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.
60) В текстовом файле **k8-48.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.
61) В текстовом файле **k8-52.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.
62) В текстовом файле **k8-55.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.
63) В текстовом файле **k8-69.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.

64) В текстовом файле **k8-75.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.
65) В текстовом файле **k8-80.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.
66) В текстовом файле **k8-84.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.
67) В текстовом файле **k8-91.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.
68) В текстовом файле **k8-94.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.
69) В текстовом файле **k8-96.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.
70) В текстовом файле **k8-97.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.
71) В текстовом файле **k8-100.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. Выведите сначала символ, из которого строится эта подцепочка, а затем через пробел – длину этой подцепочки.
72) В текстовом файле **k8-0.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. **Для каждой цепочки максимальной длины** выведите в отдельной строке сначала символ, из которого строится эта цепочка, а затем через пробел – длину этой цепочки.

73) В текстовом файле **k8-4.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. **Для каждой цепочки максимальной длины** выведите в отдельной строке сначала символ, из которого строится эта цепочка, а затем через пробел – длину этой цепочки.
74) В текстовом файле **k8-6.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. **Для каждой цепочки максимальной длины** выведите в отдельной строке сначала символ, из которого строится эта цепочка, а затем через пробел – длину этой цепочки.
75) В текстовом файле **k8-12.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. **Для каждой цепочки максимальной длины** выведите в отдельной строке сначала символ, из которого строится эта цепочка, а затем через пробел – длину этой цепочки.
76) В текстовом файле **k8-18.txt** находится цепочка из символов, в которую могут входить заглавные буквы латинского алфавита A…Z и десятичные цифры. Найдите длину самой длинной подцепочки, состоящей из одинаковых символов. Если в файле несколько подходящих цепочек одинаковой длины, нужно взять первую из них. **Для каждой цепочки максимальной длины** выведите в отдельной строке сначала символ, из которого строится эта цепочка, а затем через пробел – длину этой цепочки.
77) Текстовый файл **k8-1.txt** состоит не более чем из 10^6 символов. Определите максимальное количество идущих подряд символов, среди которых каждые два соседних различны.
78) Текстовый файл **k8-2.txt** состоит не более чем из 10^6 символов. Определите максимальное количество идущих подряд символов, среди которых каждые два соседних различны.
79) Текстовый файл **k8-3.txt** состоит не более чем из 10^6 символов. Определите максимальное количество идущих подряд символов, среди которых каждые два соседних различны.
80) Текстовый файл **k8-4.txt** состоит не более чем из 10^6 символов. Определите максимальное количество идущих подряд символов, среди которых каждые два соседних различны.
81) Текстовый файл **k8-5.txt** состоит не более чем из 10^6 символов. Определите максимальное количество идущих подряд символов, среди которых каждые два соседних различны.
82) Текстовый файл **k8-6.txt** состоит не более чем из 10^6 символов. Определите максимальное количество идущих подряд символов, среди которых каждые два соседних различны.
83) Текстовый файл **k8-7.txt** состоит не более чем из 10^6 символов. Определите максимальное количество идущих подряд символов, среди которых каждые два соседних различны.
84) Текстовый файл **k8-8.txt** состоит не более чем из 10^6 символов. Определите максимальное количество идущих подряд символов, среди которых каждые два соседних различны.
85) Текстовый файл **k8-9.txt** состоит не более чем из 10^6 символов. Определите максимальное количество идущих подряд символов, среди которых каждые два соседних различны.
86) Текстовый файл **k8-10.txt** состоит не более чем из 10^6 символов. Определите максимальное количество идущих подряд символов, среди которых каждые два соседних различны.
87) (**П.Е. Финкель, г. Тимашевск**) Текстовый файл **24-1.txt** состоит не более чем из 10^6 символов. Определите максимальное нечётное число, записанное в этом файле.

88) (**П.Е. Финкель, г. Тимашевск**) Текстовый файл **24-1.txt** состоит не более чем из 10^6 символов. Определите минимальное нечётное число, записанное в этом файле.

89) (**П.Е. Финкель, г. Тимашевск**) Текстовый файл **24-1.txt** состоит не более чем из 10^6 символов. Определите максимальное чётное число, записанное в этом файле.

90) (**П.Е. Финкель, г. Тимашевск**) Текстовый файл **24-1.txt** состоит не более чем из 10^6 символов. Определите минимальное чётное число, записанное в этом файле.

91) (**П.Е. Финкель, г. Тимашевск**) Текстовый файл **24-1.txt** состоит не более чем из 10^6 символов. Определите самое большое число, состоящее только из нечётных цифр.

92) (**П.Е. Финкель, г. Тимашевск**) Текстовый файл **24-1.txt** состоит не более чем из 10^6 символов. Определите самое большое число, состоящее только из чётных цифр.

Возрастающей подпоследовательностью будем называть последовательность символов, расположенных в порядке увеличения их номера в кодовой таблице символов ASCII.
Убывающей подпоследовательностью будем называть последовательность символов, расположенных в порядке уменьшения их номера в кодовой таблице символов ASCII.

93) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Определите длину наибольшей возрастающей подпоследовательности.

94) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-1.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Определите длину наибольшей возрастающей подпоследовательности.

95) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-2.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Определите длину наибольшей возрастающей подпоследовательности.

96) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-3.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Определите длину наибольшей возрастающей подпоследовательности.

97) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-4.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Определите длину наибольшей возрастающей подпоследовательности.

98) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе наибольшую возрастающую подпоследовательность.

99) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-1.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе наибольшую возрастающую подпоследовательность.

100) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-2.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе наибольшую возрастающую подпоследовательность.

101) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-3.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе наибольшую возрастающую подпоследовательность.

102) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-4.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе наибольшую возрастающую подпоследовательность.

103) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе

номер символа, с которого начинается наибольшая возрастающая подпоследовательность. Нумерация символов начинается с 1.

104) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-1.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе номер символа, с которого начинается наибольшая возрастающая подпоследовательность. Нумерация символов начинается с 1.

105) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-2.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе номер символа, с которого начинается наибольшая возрастающая подпоследовательность. Нумерация символов начинается с 1.

106) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-3.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе номер символа, с которого начинается наибольшая возрастающая подпоследовательность. Нумерация символов начинается с 1.

107) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-4.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе номер символа, с которого начинается наибольшая возрастающая подпоследовательность. Нумерация символов начинается с 1.

108) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Определите длину наибольшей убывающей подпоследовательности.

109) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-1.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Определите длину наибольшей убывающей подпоследовательности.

110) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-2.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Определите длину наибольшей убывающей подпоследовательности.

111) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-3.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Определите длину наибольшей убывающей подпоследовательности.

112) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-4.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Определите длину наибольшей убывающей подпоследовательности.

113) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе наибольшую убывающую подпоследовательность.

114) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-1.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе наибольшую убывающую подпоследовательность.

115) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-2.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе наибольшую убывающую подпоследовательность.

116) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-3.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе наибольшую убывающую подпоследовательность.

117) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-4.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе наибольшую убывающую подпоследовательность.

118) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе номер символа, с которого начинается наибольшая убывающая подпоследовательность. Нумерация символов начинается с 1.

119) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-1.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе номер символа, с которого начинается наибольшая убывающая подпоследовательность. Нумерация символов начинается с 1.

120) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-2.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе номер символа, с которого начинается наибольшая убывающая подпоследовательность. Нумерация символов начинается с 1.

121) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-3.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе номер символа, с которого начинается наибольшая убывающая подпоследовательность. Нумерация символов начинается с 1.

122) (**В.Н. Шубинкин, г. Казань**) Текстовый файл **24-4.txt** содержит последовательность из строчных и заглавных букв английского алфавита и цифр, всего не более 10^6 символов. Запишите в ответе номер символа, с которого начинается наибольшая убывающая подпоследовательность. Нумерация символов начинается с 1.