Feladat

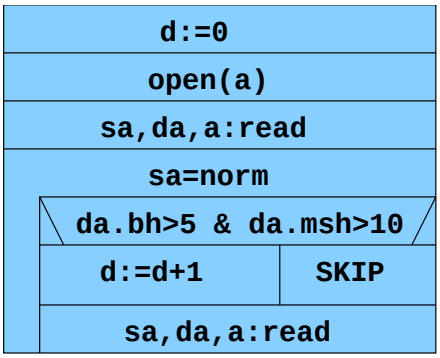
Számoljuk meg, hogy egy szöveges file hány olyan bekezdést tartalmaz, ami legalább 5 sorból áll, és a leghoszabb sora legalább 10 betűből áll. A sorok egymástól sorvége jellel vannak elválasztva. A bekezdések egymástól egy vagy több üres sorral vannak elválasztva (az üres sor csak a sorvége jelet tartalmazza).

Megoldás

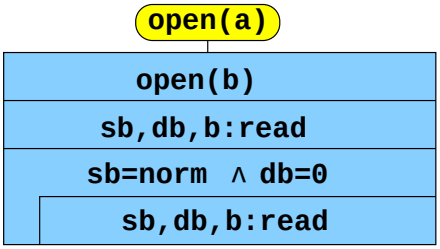
Közvetlenül a feladat nem vezethető vissza egyetlen ismert programozási tételre sem. Tegyük fel hogy az eredeti szövegfile helyett egy olyan absztrakt file-unk van ami párokat tartalmaz:egy pár az eredeti file egy bekezdése sorainak a számát és a bekezdés leghosszabb sorának a hosszát tartalmazza. Ekkor a feladat visszavezethető a számlálás tételére. Legyen az eredeti file `x:F` (amiből karaktereket olvasunk), és legyen az absztrakt file `a:A` (amiből a bekezdések sorainak a számát és a leghoszabb sorának a hosszát olvassuk). Az `a`-beli elemek rekordtípusa két számot tartalmaz: bekezdés sorainak száma (`bh`) és a bekezdés maximális sorhossza (`msh`).

Absztrakt program

Az absztrakt `a:A` file-on a számlálás:



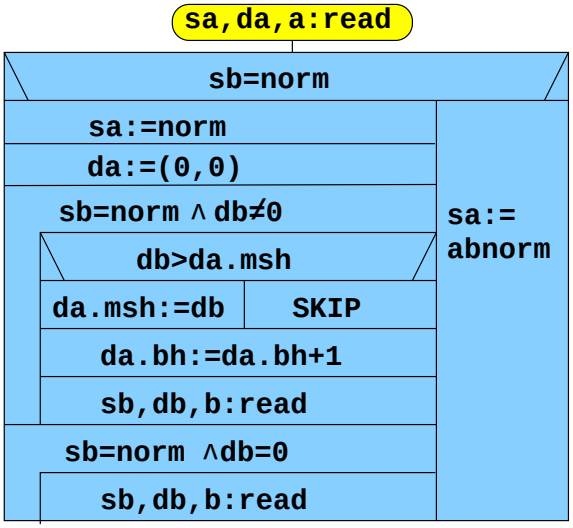
Hátra van még az absztrakt file típusműveleteinek megvalósítása. A problémát az okozza, hogy az eredeti file-t olvasva nehéz a bekezdés sorainak számát és maximális sorhosszát előállítani. Alkalmazzunk ismét adatabsztrakciót: tegyük fel, hogy az eredeti file helyett egy olyan file-unk van, ami sorhosszokat tartalmaz. Legyen ez az absztrakt file `b:B`. Ennek a file-nak a felhasználásával az `a`-beli műveletek megvalósítása egyszerű. Az `open` beolvassa az első bekezdés első sorának hosszát:

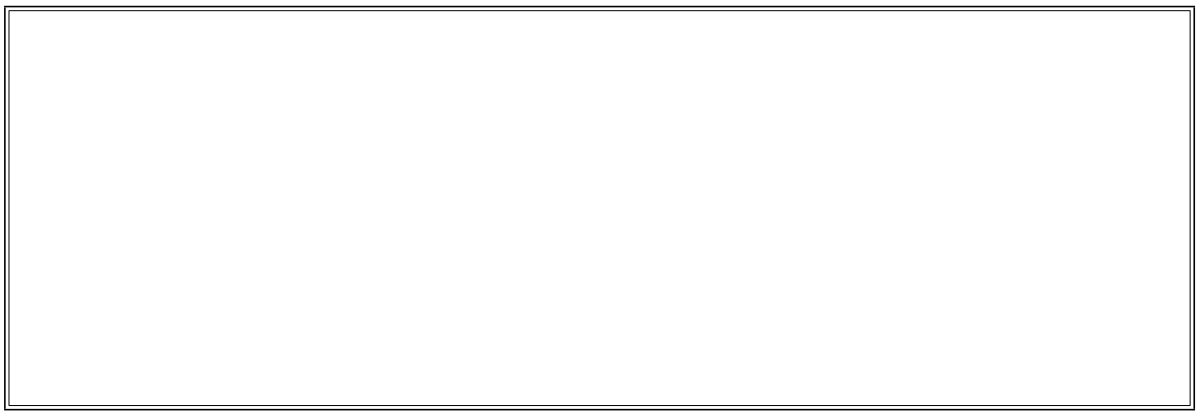


Az `read` művelet a bekezdés sorhosszai alapján kiszámítja az alábbi rekurzív formulával definiált függvényt:

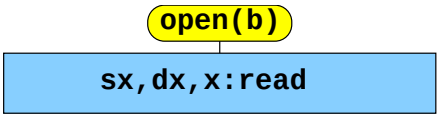
$$\begin{aligned} f(0) &= (0,0) \\ f(i+1) &= \begin{cases} f_1(i)+1, b_{i+1} & \text{ha } b_{i+1} > f_2(i) \\ f_1(i)+1, f_2(i) & \text{különben} \end{cases} \end{aligned}$$

és rááll az következő bekezdés elejére:

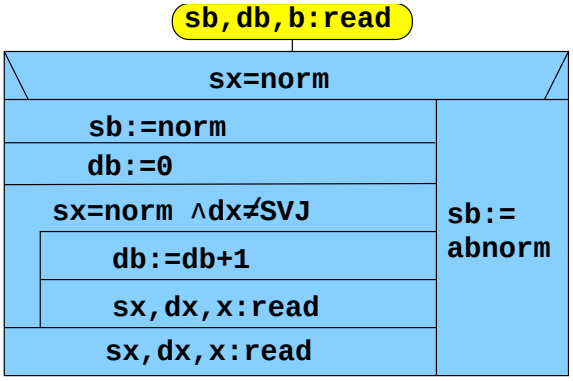




A feladat megoldásából még hátra van a `b` absztrakt file implementációja. A `b`-ből sorokat olvasunk, az `open()` beolvassa az első sor első karakterét:



A `read()` művelet elolvassa sort és megszámolja a karaktereket, majd beolvassa a következő sor első karakterét:



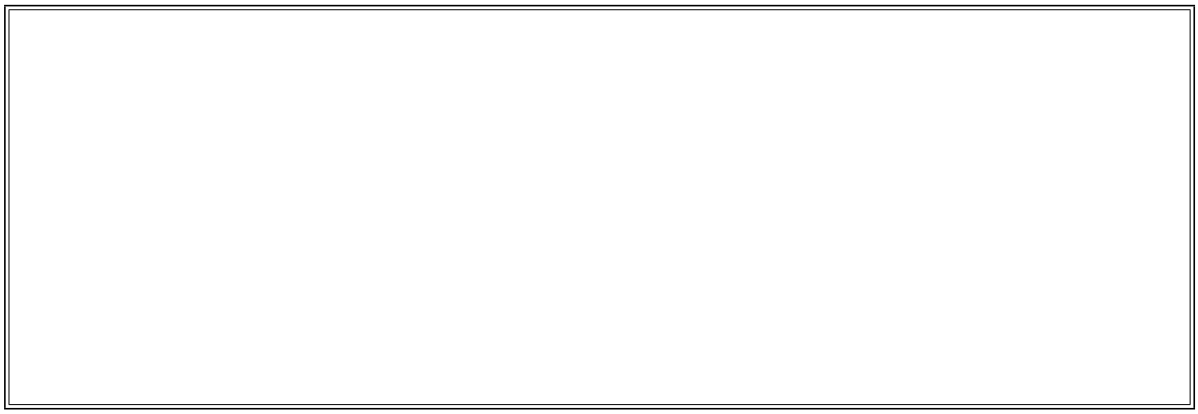
Megoldás C++-ban

Lévén az A és a B egy-egy absztrakt file típus, C++-ban osztállyal fogjuk megvalósítani. A B reprezentációjában az eredeti x file, valamint a műveletek során használt (és a műveletek között megosztott) sx és dx változók szerepelnek. A típusnak (a konstruktoron kívül) két művelete lesz: az tt open és a `read`.

```
#include <iostream>
#include <fstream>
using namespace std;

enum status {norm, abnorm};

const char SVJ='\n';

class B{
   ifstream x;
   char     dx;
   status   sx;
public:
   B(const char* fn);
   void open();
   void read(status& sb, int& db);
};
```

A műveletek implementációja:

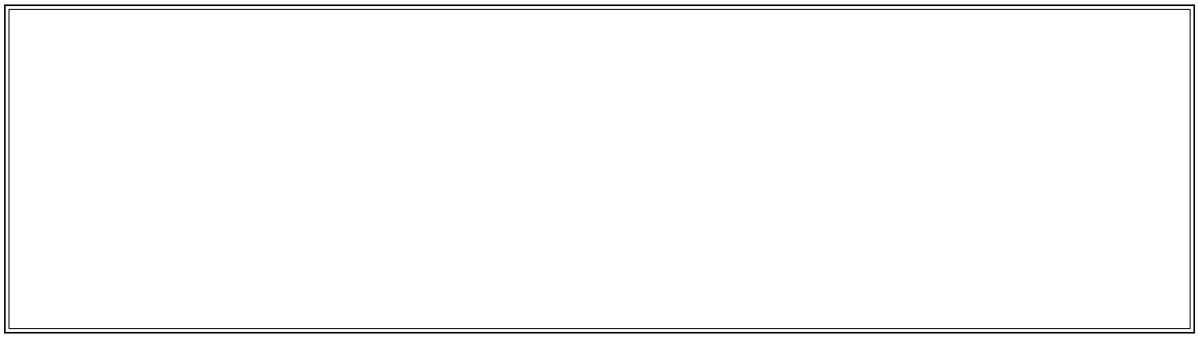
```
B::B(const char* fn):
  x(fn),dx(0),sx(norm)
{
}

void B::open()
{
   x.get(dx);
   sx = x.eof() ? abnorm: norm;
}

void B::read(status& sb, int& db)
{
   if(sx == norm){
      sb=norm;
      db=0;
      while(sx==norm && dx!=SVJ){
         db++;
         x.get(dx);
         sx = x.eof() ? abnorm: norm;
      }
      x.get(dx);
      sx = x.eof() ? abnorm: norm;
   }else{
      sb=abnorm;
   }
}
```



Az `A` osztály reprezentációjában egy `b:B` file, valamint a műveletek során használt (és a műveletek között megosztott) `sb` és `db` változók szerepelnek. A típusnak (a konstruktoron kívül) két művelete lesz: az tt open és a `read`.

```
struct Rec{
   int bh;
   int msh;
};

class A{
   B        b;
   int      db;
```

```
   status   sb;
public:
   A(const char* fn);
   void open();
   void read(status& sa, Rec& da);
};
```



A műveletek implementációja:

```
A::A(const char* fn):
  b(fn),db(0),sb(norm)
{
}

void A::open()
{
   b.open();
   b.read(sb,db);
   while(sb==norm && db==0){
      b.read(sb,db);
   }
}

void A::read(status& sa, Rec& da)
{
   if(sb == norm){
      sa=norm;
      da.bh=0;
      da.msh=0;
      while(sb==norm && db!=0){
         da.bh++;
         if(da.msh < db){
            da.msh=db;
         }
         b.read(sb,db);
      }
      while(sb==norm && db==0){
         b.read(sb,db);
      }
   }else{
      sa=abnorm;
   }
}
```

Az A osztály ismeretében a főprogram implementációja:

```
int main(int argc, char *argv[])
{
  int d = 0;
  A   a("x.txt");
  Rec da;
  status sa;

  a.open();
  a.read(sa, da);
  while(sa==norm){
     if(da.bh > 5 && da.msh > 10){
        d++;
     }
     a.read(sa, da);
  }
  cout << d << endl;
  return 0;
}
```