

Feladat

Gyűjtsük ki egy n elemű, szavakat tartalmazó vektorból a „G” betűvel kezdődő szavakat!

Specifikáció

A feladatnak egy általános, tetszőleges szűrési feltétellel működő változatát fogalmazzuk meg:

$$\begin{aligned}
 A &= \mathbb{V}_{Szó} \times (Szó \rightarrow \mathbb{L}) \times \mathbb{S}_{Szó} \quad (\times \mathbb{Z}) \\
 &\quad x \quad \quad \quad \pi \quad \quad \quad y \quad \quad \quad k \\
 B &= \mathbb{V}_{Szó} \times (Szó \rightarrow \mathbb{L}) \\
 &\quad x' \quad \quad \quad \pi' \\
 Q &= x' = x \wedge \pi' = \pi \\
 R &= y = seq(x|Szó_\pi),
 \end{aligned}$$

ahol

$$Szó_\pi := \{u \in Szó \mid \pi(u)\}$$

Absztrakt program

A feladatot az elemenkénti feldolgozás tételének vektorokra vonatkozó változatával oldjuk meg, az alábbi függvény segítségével:

$$f(u \in Szó) := \begin{cases} \{u\}, & \text{ha } \pi(u) \\ \emptyset, & \text{ha } \neg\pi(u) \end{cases}$$

Tehát az absztrakt programunk:

$y := \langle \rangle$							
$k := x.lob - 1$							
$k \neq x.hib$							
<table border="1" style="width: 100%; border-collapse: collapse; text-align: center;"> <tr> <td colspan="2" style="border: none;">$\pi(x_{k+1})$</td> </tr> <tr> <td style="border: none;">$y : hiext(x_{k+1})$</td> <td style="border: none;">SKIP</td> </tr> <tr> <td colspan="2" style="border: none;">$k := k + 1$</td> </tr> </table>		$\pi(x_{k+1})$		$y : hiext(x_{k+1})$	SKIP	$k := k + 1$	
$\pi(x_{k+1})$							
$y : hiext(x_{k+1})$	SKIP						
$k := k + 1$							

Fekete doboz-tesztelés

1. Üres vektor
2. Csak $\neg\pi$ tulajdonságú szavakat tartalmazó vektor
3. Csak π tulajdonságú szavakat tartalmazó vektor
4. π és $\neg\pi$ tulajdonságú szavakat egyaránt tartalmazó vektor

C++ implementáció

```
#include <string>
#include <vector>
#include <list>

typedef std::string      Word;

struct Vector_Word
{
    int    size;
    Word* data;
};

typedef std::list<Word>  Sequence_Word;
typedef bool             (Predicate) (const Word&);

void Filter(const Vector_Word &x, Predicate pi, Sequence_Word &y)
{
    y.clear ();

    for (int i = 0; i != x.size; ++i)
        if (pi(x.data[i]))
            y.push_back (x.data[i]);
}

bool Begins_with_G(const Word &word)
{
    return word[0] == 'G';
}

#include <iostream>

void Process (std::istream &stream)
{
    Vector_Word x;

    stream >> x.size;
    x.data = new Word[x.size];

    for (int i = 0; i != x.size; ++i)
        stream >> x.data[i];

    Sequence_Word y;
    Filter(x, Begins_with_G, y);

    for (Sequence_Word::const_iterator i = y.begin(); i != y.end(); ++i)
        std::cout << *i << std::endl;

    delete [] x.data;
}

#include <fstream>

void Process (const std::string &filename)
{
    if (filename == "-")
    {
```

```
        Process (std::cin);
    } else {
        std::ifstream stream (filename.c_str ());
        if (stream.fail ())
            std::cerr << "Opening_" << filename << "_failed." << std::endl;
        else
            Process (stream);
    }
}

int main (int argc, char **argv)
{
    if (argc > 1)
        for (int i = 1; i != argc; ++i)
            Process (argv[i]);
    else
    {
        std::cout << "Adatbeolvasas_modja:" << std::endl
                  << "1_-Szoveges_file" << std::endl
                  << "2_-Stdin" << std::endl
                  << std::endl;

        int v;
        std::cin >> v;
        if (v == 1)
        {
            std::string filename;
            std::cout << "Filenev:";
            std::cin >> filename;
            Process (filename);
        }
        else if (v == 2)
        {
            Process (std::cin);
        }
        else
        {
            std::cerr << "HIBA! Ervenytelen_beviteli_mod." << std::endl;
            exit (-1);
        }
    }

    return 0;
}
```

Fehér doboz-tesztelés

1. Beolvasás file-ból, ill. stdin-ről
2. Input utolsó sorának végén újsor-karakter szerepel/nem szerepel
3. Nemlétező file megadása