7 stycznia 2016

Zadanie 9

Nie używając klasy **string** (z nagłówka *<string>*), zdefiniować własną klasę **String**, której obiekty przechowują napisy (wskaźniki do tablic znaków) o dowolnej długości. Można używać funkcji w rodzaju **strlen** i **strcpy** z nagłówka *<cstring>* oraz funkcji operujących na znakach (jak **tolower** czy **toupper** z nagłówka *<cctype>*.
Zdefiniować odpowiednie konstruktory (w szczególności kopiujący) i destruktor. Zdefiniować też metody tak, że po `String s("New York");`

- `s.toUpper()` zmienia napis na `"NEW YORK"` i zwraca odniesienie (referencję) do `s` po tej zmianie;
- `s.toLower()` zmienia napis na `"new york"` i zwraca odniesienie (referencję) do `s` po zmianie;
- `s.length()` zwraca długość napisu (nie licząc ewentualnego znaku NUL na końcu).

Przeciążyć też operatory

- '<<' tak, aby możliwe było wyprowadzanie napisów do strumienia wyjściowego;
- '==' i '!=', by `s==t` i `s!=t` działały jak `s.equals(t)` i `!s.equals(t)` w Javie. Powinno też działać porównanie z C-napisami (np. `s != "York"`);
- przypisania '=';
- dodawania '+' (konkatenacji) obiektów klasy **String**, również do zwykłych C-napisów, np.: `s="abc"+s1+s2+"def"`.

Na przykład poniższy program

```
#include <iostream>
#include <cstring>
#include <cctype>
using namespace std;

class String {
    char* nap;
public:
    String(const char* n);
    String(const String& s);
    String operator+(const String& s) const;
    String operator+(const char* n) const;
    String& operator=(const String& s);
    bool operator==(const String& s) const;
    bool operator!=(const String& s) const;
    String& toLower();
```

```
    String& toUpper();
    size_t length() const;
    ~String();

    friend String  operator+(const char*,const String&);
    friend ostream& operator<<(ostream&, const String&);
};

int main(void) {
    String s = "To " + String("be ") + "or not to be";
    cout << s << endl;
    if (s == "To be or not to be")
        s = String(s.toUpper());
    else
        s = String(s.toLower());
    cout << "Length = " << s.length() << endl;
    cout << s << endl;
}
```

po uzupełnieniu implementacji powinien się skompilować i wypisać

```
To be or not to be
Length = 18
TO BE OR NOT TO BE
```

Termin: do 22 stycznia (włącznie)

---

*Rozwiązania, w postaci **jednego** pliku źródłowego zawierającego treść programu, proszę wrzucać w systemie EDU do katalogu „Foldery zadań / Zadanie_XX", gdzie 'XX' jest numerem zadania.*

Nazwą pliku powinno być nazwisko z dużej litery (bez polskich znaków); rozszerzeniem musi być '`.cpp`', czyli np. `Malinowska.cpp`.