Egzamin z Zaawansowanego Programowania Komputerowego

25 czerwca 2020

1. (10 pkt.) Napisać funkcję

```
bool a(string s);
```

która zwraca true jeśli napis zawiera poprawny układ nawiasów, tj.
- wszystkie znaki w s to '(' albo ')'
- każdy nawias jest zamknięty.

Np. a("()(()())")=true, a("(()")=false (jeden z nawiasów nie jest zamknięty), a(")(")= false (nawias jest zamknięty przed otwarciem, a("())(()")=false (jak poprzednio).

2. (15 pkt.) Rozważamy dwuosobowa grę. Gracze mają do dyspozycji dwie liczby całkowite dodatnie a i b. Gracze wykonują ruchy na zmianę. Można wykonywać ruchu jednego z dwóch rodzajów:
- dodać 1 do jednej z liczb a, b (np. (a=5, b=8) => (a=5, b=9))
- zastąpić większą z tych liczb przez jej resztę z dzielenia przez mniejszą liczbę (np. (a=5, b=8) => (a=5, b=3)).

Wygrywa gracz, który doprowadzi do wyzerowania jednej z liczb. Napisać funkcję

```
bool w(int a, int b);
```

która zwraca true wtedy i tylko wtedy, gdy gracz rozpoczynający grę wygra jeśli obaj gracze będą grali optymalnie. Np.
- w(6,3)=true (wystarczy wykonać ruch (6,3)=>(0,3)),
- w(3,2)=false (możemy wykonać ruchy do (4,2), (3,3) lub (1,2), ale z każdego przeciwnik wygrywa w jednym ruchu)

3. (10 pkt.) Dana jest funkcja

```
int c(int n) {
      if(n<=0) return 0;
      if(n%2==1) return c(n-1)+n;
      return c(n-1);
}
```

Napisać równoważną funkcję, która nie używa rekurencji i podać jej złożoność.

4. (15 pkt.) Dana jest struktura

```
class ElListy{public: int wartosc; ElListy* nast; };
```

której będziemy używać do tworzenia list jednokierunkowych.
Napisać funkcję

```
ElListy* sortuj(ElListy* a);
```

która przestawia elementy listy a tak, aby stanowiły ciąg niemalejący. Należy użyć istniejących już elementów. Funkcja powinna zwrócić wskaźnik do nowej, przestawionej listy.

5. (10 pkt.) Dana jest struktura

```
class Wezel{ public: int wartosc; Wezel* lewy; Wezel* prawy; };
```

której będziemy używać do tworzenia drzew.
Napisać funkcję

```
int min(Wezel* d);
```

która zwraca najmniejszą wartość znajdującą się w drzewie `d`.

6. (20 pkt.) Stworzyć klasę `Mapa`, której obiekty reprezentują mapy pewnego kraju. Na mapie zaznaczone są miasta i drogi łączące te miasta. Parę miast może łączyć co najwyżej jedna droga. Drogi są dwukierunkowe. Każdego miasto ma inną nazwę.

```
class Mapa {
public:
      Mapa();
      Mapa(vector<string>& listaMiast);
      void dodajMiasto(string nazwa);
      void dodajDroge(string miasto1, string miasto2, double dlugosc);
      double odleglosc(string miasto1, string miasto2);
      vector<string> sasiedzi(string miasto);
```

```
};
```

Opis metod publicznych, które należy napisać:

```
Mapa();
```

Tworzy pustą mapę.

```
Mapa(vector<string>& listaMiast);
```

Tworzy mapę, na której znajdują się miasta podane w parametrze, nie ma żadnych dróg.

```
void dodajMiasto(string nazwa);
```

Dodaje miasto o podanej nazwie (jeśli go jeszcze nie ma).

```
void dodajDroge(string miasto1, string miasto2, double dlugosc);
```

Dodaje drogę o podanej długości pomiędzy miastami. Jeśli miasto lub miasta nie istnieją, nic się nie dzieje. Jeśli droga pomiędzy podanymi miastami już istnieje, zostaje zmieniona jej długość.

```
double odleglosc(string miasto1, string miasto2);
```

Zwraca długość drogi (bezpośredniej) pomiędzy miastami lub -1 jeśli droga bądź któreś z miast nie istnieją.

```
vector<string> sasiedzi(string miasto);
```

Zwraca tablicę miast połączonych bezpośrednio drogą z danym miastem. Jeśli miasto nie istnieje, zwraca pusty wektor.