Kolokwium z Zaawansowanego Programowania Komputerowego

28 kwietnia 2021
Grupa A (15:00)

A1. (5 pkt.) Napisać funkcję

```
bool a(int** t, int n);
```

która zwraca true jeśli wszystkie wiersze i kolumny tablicy t są posortowane rosnąco. Tablica t ma rozmiar n x n.
Oszacować złożoność napisanej funkcji w zależności od n.

A2. (5 pkt.) Dana jest funkcja

```
int f(int n) {
    if(n<3) return n;
    else return f(n-3)+f(n-1);
}
```

Oszacować tej funkcji złożoność w zależności od n.
Napisać równoważną funkcję bez użycia rekurencji.

A3. (4 pkt.) Dana jest klasa

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

Napisać funkcję

```
bool petla(ElListy* lista);
```

która zwraca false jeśli lista się kończy a true jeśli lista się zapętla, tj. przechodząc listę nigdy nie dojdziemy do końca (wtedy elementy listy zaczną się w pewnym momencie powtarzać).

A4. (6 pkt.) Stworzyć klasę Tablica, której obiekty reprezentują tablice liczb typu double indeksowane dowolnym przedziałem liczb a..b.
Uwaga: zakładamy, że a<=b, ale nie zakładamy, że te liczby są dodatnie!
Metody publiczne:

`Tablica(int a, int b);`
: tworzy tablicę z elementami t[a] ... t[b] wypełnioną zerami

`void ustaw(int i, double w);`
: ustawia element o indeksie i na w, jeśli i jest poza zakresem a...b nic się nie dzieje

`double czytaj(int i);`
: zwraca element o indeksie i; jeśli i jest poza zakresem zwraca 0.

`Tablica podtablica(int c, int d);`
: zwraca nową tablicę {t[c]...t[d]}, elementy poza zakresem [a,b] ustawiamy na 0.

`void przesun();`
: przesuwa indeksy w tablicy o 1 miejsce do góry, np. tablica {t[2]=1, t[3]=4; t[4]=3} zamienia się w tablicę {t[3]=1, t[4]=4; t[5]=3}

Kolokwium z Zaawansowanego Programowania Komputerowego

28 kwietnia 2021
Grupa B (18:30)

B1. (5 pkt.) Napisać funkcję

```
bool b(vector<int>& v);
```

która zwraca true wtedy i tylko wtedy, gdy każdy element wektora v jest większy od sumy elementów go poprzedzających. Np.
b([1,2,6,20])=true (bo 20>1+2+6, 6>1+2, 2>1, 1>0)
b([3,4,7])=false (bo 7 nie jest większe niż 3+4).
Suma elementów poprzedzających v[0] wynosi oczywiście 0.

B2. (5 pkt.) Dana jest funkcja

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

Oszacować tej funkcji złożoność w zależności od n.
Napisać równoważną funkcję bez użycia rekurencji.

B3. (4 pkt.) Dana jest klasa

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

Napisać funkcję

```
ElListy* przestaw(ElListy* lista);
```

która przestawia pierwszy element listy na koniec. Należy zwrócić wskaźnik do pierwszego elementu nowej listy powstałej po przestawieniu.

B4. (6 pkt.) Zaimplementować klasę Slownik, której obiekty zawierają pary słów: polskie i jego angielskie tłumaczenie. Zakładamy, że każde polskie słowo ma dokładnie jeden angielski odpowiednik (choć różne polskie słowa mogą mieć takie samo tłumaczenie na angielski).
Metody publiczne:

`Slownik();`
tworzy pusty słownik.
`void dodaj(string pol, string ang);`
dodaje polskie słowo pol wraz z tłumaczeniem angielskim ang. Jeśli pol jest już w słowniku, tłumaczenie jest zastępowane.
`void usun(string pol);`
usuwa polskie słowo wraz z tłumaczeniem.
`string polToAng(string pol);`
zwraca angielskie tłumaczenie podanego słowa lub napis pusty jeśli słowa nie ma w słowniku.
`string angToPol(string ang)`
zwraca jakiekolwiek tłumaczenie angielskiego słowa ang na polski lub napis pusty jeśli żadnego nie ma.