Zaawansowane programowanie komputerowe

Zadania przygotowawcze do kolokwium

A1. Napisać funkcję, która zwraca wartość true wtedy i tylko wtedy, gdy tablica t o długości n jest uporządkowana (rosnąco lub malejąco).

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

A2. Napisać funkcję, która zwraca medianę elementów z tablicy t o rozmiarze n, tj. dowolną liczbę m taką, że zbiory {i: t[i]>m} i {i:t[i]<m} są równoliczne.

```
int f(int* t, int n);
```

A3. Napisać funkcję

```
int f(int a, int b, int n);
```

która zlicza liczbę sposobów, na jakie można zapisać liczbę n w postaci sumy liczb a i b (identyfikujemy różne kolejności składników)

A4. Napisać funkcję

```
bool f(char* a, char* b);
```

która zwraca true wttw jeden napis jest przesunięciem drugiego. Np. przesunięciami napisu "ABCD" są napisy "BCDA", "CDAB", "DABC" i oczywiście "ABCD".

A5. Napisać funkcję

```
int f(int*t, int rozmiar);
```

która zwraca długość najdłuższego ciągu kolejnych elementów tablicy t tworzących ciąg arytmetyczny.

B1. Dana jest funkcja

```
int f(int n) {
	if(n<3) return 1;
	return f(n-1)+f(n-2)+f(n-3);
}
```

a. Obliczyć przybliżoną złożoność tej funkcji ze względu na parametr n.
b. Napisać równoważną funkcję nierekurencyjną.

B2. Dana jest funkcja

```
int f(int a, int b) {
	if(a<=0 || b<=0) return 1;
	return f(a-1, b)+f(a,b-1)+a*b;
}
```

a. Obliczyć przybliżoną złożoność tej funkcji.
b. Napisać równoważną funkcję nierekurencyjną.

B3. Napisać funkcję, która wypisuje na ekran wszystkie możliwe przedstawienia liczby n w postaci nierosnącej sumy liczb całkowitych dodatnich. Oszacować złożoność tej funkcji.

C1. Stworzyć klasę Tablica przechowującą elementy typu double.

```
class Tablica {
public:
	Tablica(int n, double x); // tworzy nową tablicę wypełnioną
elementami x
	void ustaw(int i, double v); // ustawia wartość o wskazanym
indeksie
	double wartosc(int i); // zwraca wartość o podanym indeksie
	Tablica& dolacz(Tablica& t); // dołącza do tablicy elementy z
tablicy będącej parametrem
```

```
	int rozmiar(); // zwraca rozmiar tablicy
};
```

C2. Stworzyć klasę Czas, która przechowuje aktualny czas (godzinę i minutę)

```
class Czas {
public:
	Czas();
	Czas(int h, int m);
	Czas& dodajGodziny(int ileGodzin);
	Czas& dodajMinuty(int ileMinut);
	void drukuj(); // drukuje godzinę w formacie 12-godzinnym (np.
7:30AM lub 1:14PM)
	int zaIleMinut(Czas t); // podaje, za ile minut nastąpi podany
czas
};
```

C3. Napisać klasę ZbiorNapisow, która przechowuje zbiory napisów

```
class ZbiorNapisow {
public:
	ZbiorNapisow(); // tworzy pusty zbiór
	void dodaj(char* napis); // dodaje napis
	void usun(char* napis); // usuwa napis
	bool nalezy(char* napis); // sprawdza, czy podany napis należy do
zbioru
	int ile(); // zwraca liczbę napisów
	double sredniaDlugosc(); // zwraca średnią długość napisu ze
zbioru
};
```

C4. Labirynt to prostokątny budynek podzielony na kwadratowe pomieszczenia jednakowej wielkości. Pomieszczenia są ponumerowane parami liczb całkowitych (a,b), 0<=a<s, 0<=b<w, gdzie s i w są rozmiarami labiryntu. Pomiędzy sąsiednimi pomieszczeniami może znajdować się przejście, pomieszczenia skrajne nie mogą mieć wyjść na zewnątrz. Napisać klasę Labirynt, której obiekty reprezentują labirynty opisane powyżej.

```
class Labirynt {
public:
	Labirynt(int s, int w); // tworzy nowy Labirynt o rozmiarach s,w
bez przejść pomiędzy polami
	void wstawPrzejscie(int x, int y, char k); // wstawia przejście z
pomieszczenia (x,y) w kierunku k
	void usunPrzejscie(int x, int y, char k); // usuwa przejście z
pomieszczenia (x,y) w kierunku k
	bool jestPrzejscie(int x, int y, char k); // zwraca informację
czy jest przejście
}
```