Zaawansowane programowanie komputerowe

Zadania przygotowawcze do egzaminu

A. Zdefiniowana jest struktura reprezentująca element listy

```
class Element {public: int wartosc; Element* nastepny; Element*
poprzedni; };
```

1. Napisać funkcję

```
int dlugosc(Element* lista);
```

zwracającą długość listy. Parametr lista wskazuje na pierwszy element listy (tj. taki, że nie mam poprzedniego).

2. Napisać funkcję

```
bool posortowana(Element* lista);
```

zwracającą true wtedy i tylko wtedy, gdy lista jest posortowana.

3. Napisać funkcję

```
Element* usun(int w);
```

która usuwa wszystkie elementy listy o wartości równej w. Należy zwrócić wskaźnik do listy z usuniętymi elementami.

B. Dana jest klasa

```
class Graf {
public:
      Graf(int n); // tworzy graf o n wierzchołkach ponumerowanych 1..n
      void polacz(int i, int j); // łączy wierzchołki nr i oraz j
      int liczbaKrawedzi(int i); // zwraca liczbę krawędzi wychodzących
z wierzchołka i
      int* listaKrawedzi(int i); // zwraca tablicę z numerami krawędzi
wychodzących z wierzchołka i
};
```

1. Napisać funkcję,

```
bool jestTrojkat(Graf& g);
```

która zwraca true, jeśli w grafie g istnieją trzy wierzchołki połączone wzajemnie ze sobą

2. Napisać funkcję

```
bool spójny(Graf& g)
```

która zwraca true jeśli każda parę wierzchołków g można połączyć ciągiem krawędzi.

3. Napisać funkcję

```
Graf suma(Graf& g, Graf& h)
```

która zwraca graf będący sumą grafów g i h.

C. Stworzyć klasę OkraglyStol, której obiekty reprezentują osoby siedzące przy okrągłym stole. Każda osoba reprezentowana jest przez jej numer (int), przy stole może znajdować się tylko jedna osoba o danym numerze.

```
class OkraglyStol {
public:
      OkraglyStol();
      void posadz(int n); // posadz osobę o numerze n (gdziekolwiek) o
ile jeszcze jej nie ma
      void posadz(int n, int k); // posadz osobe o numerze n po lewej
stronie osoby o numerze k (jeśli jej nie ma)
```

```
      void usun(int n); // usuwa osobę o zadanym numerze
      bool jestOsoba(int n); // zwraca true jeśli osoba jest przy stole
      int poPrawej(int n); // zwraca numer osoby siedzącej po prawej
stronie osoby o numerze n (można założyć, że osoba jest przy stole)
      int ile(); // zwraca liczbę osób
};
```

D. Stworzyć klasę Hierarchia, reprezentującą zbiór osób (reprezentowanych przez liczby typu int) oraz zależności służbowe pomiędzy nimi. Każda osoba (poza jedną) posiada szefa (inną osobę).

```
class Hierarchia {
public:
      Hierarchia(int n); // tworzy nową hierarchię zawierającą jednyną
osobę n
      void dodaj(int n, int s); // dodaje nową osobę n, jej szefem
będzie s (jeśli nie ma s lub n już jest, nic nie robi)
      void usun(int n); // usuwa n z hierarchii, podwładni n stają się
podwładnymi szefa n (jeśli n nie ma szefa, nic nie robi)
      int szef(int n); // zwraca szefa osoby n (jeśli n nie ma szefa,
zwraca n)
      int liczbaPodwladnych(int n); // zwraca liczbę podwładnych osoby n
};
```