KOLOKWIUM II

ZESTAW D

2025-01-22

1. (7 p.) Dana jest następująca deklaracja klasy `Element`, przechowująca elementy mające być umieszczone w drzewie BST (Binary Search Tree) i przechowująca jako klucz liczbę rzeczywistą `Key`:

```
class Element {
    private:
        Element *Parent;
        Element *Left;
        Element *Right;
        double Key;
    public:
        void SetParent(Element *el);
        void SetLeft(Element *el);
        void SetRight(Element *el);
        Element *GetParent();
        Element *GetLeft();
        Element *GetRight();
        void SetKey(double wart);
        double GetKey();
};
```

Metody klasy Element: `SetParent`, `SetLeft` i `SetRight` ustawiają na zadany wskaźnik odpowiednio pola `Parent`, `Left` i `Right`, zaś metody `GetParent`, `GetLeft` i `GetRight` zwracają odpowiednio wskaźniki `Parent`, `Left` i `Right`. Metoda `SetKey` ustawia wartość klucza na `wart`, a metoda `GetKey` zwraca wartość klucza przechowywanego w danym obiekcie.

Zadeklaruj klasę BST pozwalającą na przechowywanie obiektów klasy `Element` w drzewie BST. Zdefiniuj metody `Insert` wstawiające element o wskaźniku do niego zadanym w argumencie tej metody do drzewa; oraz metodę `Successor` odszukującą element o następnej (względem klucza węzła przekazanego przez wskaźnik w argumencie tej funkcji) wartości klucza w tym drzewie.

2. (7 p.) Dana jest następująca deklaracja klasy `Element`, przechowująca elementy mające być przechowywane w kolejce i przechowująca jako klucz znak `Key`:

```
class Element {
    private:
        Element *next;
        Element *prev;
        char Key;
    public:
        void SetNext(Element *el);
        void SetPrev(Element *el);
        Element *GetNext();
        Element *GetPrev();
        void SetKey(char zn);
        char GetKey();
};
```

Metody klasy `Element`: `SetNext` i `SetPrev` ustawiają na zadany wskaźnik odpowiednio pola `next` i `prev`, zaś metody `GetNext` i `GetPrev` zwracają odpowiednio wskaźniki `next` i `prev`. Metoda `SetKey` ustawia wartość klucza na `zn`, a metoda `GetKey` zwraca wartość klucza przechowywanego w danym obiekcie.

Zadeklaruj klasę `Kolejka` pozwalającą na przechowywanie obiektów klasy `Element` w kolejce. Zdefiniuj metody `Detach` usuwającą element z początku kolejki; oraz metodę `EmptyQueue` zwracającą informację, czy kolejka jest pusta.

3. (10 p.) Dane są następujące deklaracje:

```
class Element {
    private:
        Element *next;
        Element *prev;
        // ...
    public:
        // ...
        void SetNext(Element *el);
        void SetPrev(Element *el);
        Element *GetNext();
```

```
        Element *GetPrev();
};

class Lista {
    private:
        Element *head;
        Element *tail;
    public:
        // ...
        void Przesun(Element *x, int oile);
};
```

Metody klasy `Element`: `SetNext` i `SetPrev` ustawiają na zadany wskaźnik odpowiednio pola `next` i `prev`, zaś metody `GetNext` i `GetPrev` zwracają odpowiednio wskaźniki `next` i `prev`. Zapisz treść metody `Przesun`, która dla zadanego parametru `x`, będącego wskaźnikiem do elementu listy (zakładamy, że ten element znajduje się na liście), oraz liczby całkowitej `oile` dokona przesunięcia elementu `x` na liście o `oile` miejsc w kierunku końca listy. Jeśli wartość `oile` jest dodatnia `x` powinno być przesunięte w kierunku końca listy o taką liczbę miejsc, jeśli ujemne o powinno być przesunięte w kierunku początku listy o liczbę miejsc równą |`oile`|. Jeśli `oile` jest równe 0, element nie powinien być przesuwany. Jeśli wartość bezwzględna `oile` jest większa niż liczba elementów listy w danym kierunku, to element powinien znaleźć się na końcu listy jeśli `oile` było dodatnie, albo na początku listy, jeśli `oile` było ujemne. Metoda ta powinna być napisana tak, aby spójność listy została zachowana.