Cel zajęć. Celem zajęć jest zapoznanie się ze sposobem działania popularnych kolekcji.

Wprowadzenie teoretyczne. Rozważana w ramach niniejszych zajęć tematyka jest ważna, gdyż kolekcje są powszechnie używane do przechowywania danych i efektywnego zarządzania danymi. Aby ze zrozumieniem zrealizować zadania przewidziane do wykonania w ramach zajęć laboratoryjnych, należy znać znaczenie pojęcia kolekcji oraz cechy następujących kolekcji: listy jednokierunkowej, listy dwukierunkowej, kolejki, stosu.

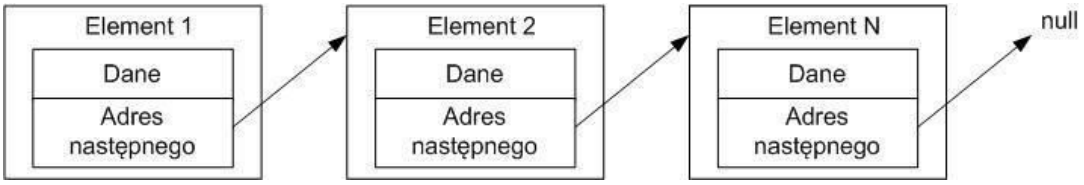
1. Kolekcja

Jest to pojęcie stosowane w odniesieniu do klas, które umożliwiają przechowywanie oraz efektywne przetwarzanie obiektów danego typu. Jedną z największych zalet kolekcji jest możliwość dynamicznej zmiany ich rozmiaru – liczba przechowywanych przez kolekcje elementów nie jest ograniczona w momencie utworzenia danej kolekcji, tak jak ma to miejsce w przypadku tablicy.

2. Lista

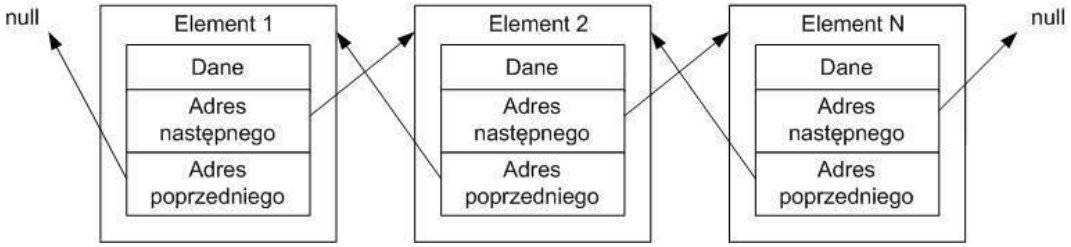
Jest kolekcją składającą się z uporządkowanych liniowo elementów. Lista umożliwia wstawianie elementów w miejsce o dowolnym indeksie. Ta sama zasada dotyczy pobierania elementu z listy. Do prawidlowego manipulowania listą wymagane jest przechowywanie przez listę adresu jej pierwszego elementu.

Lista jednokierunkowa to lista, w której każdy element wskazuje na element następny.



Rys. 1 – Lista jednokierunkowa

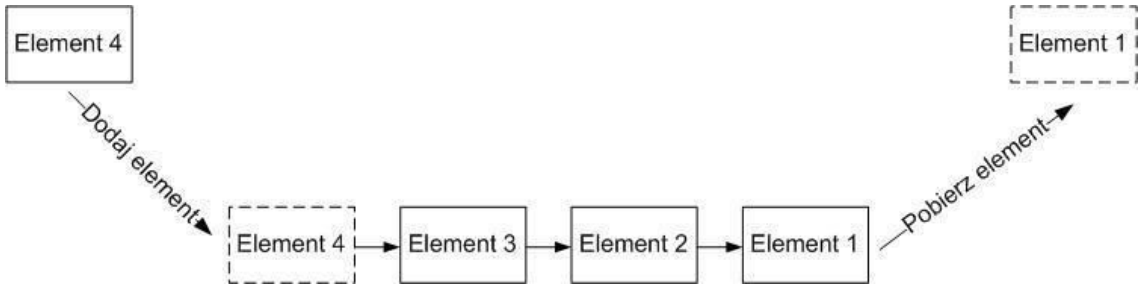
Lista dwukierunkowa to lista, w której każdy element wskazuje na element następny i na element poprzedni. Listy dwukierunkowe umożliwiają łatwiejszy dostęp do danych, niż listy jednokierunkowe.



Rys. 2 – Lista dwukierunkowa

3. Kolejka

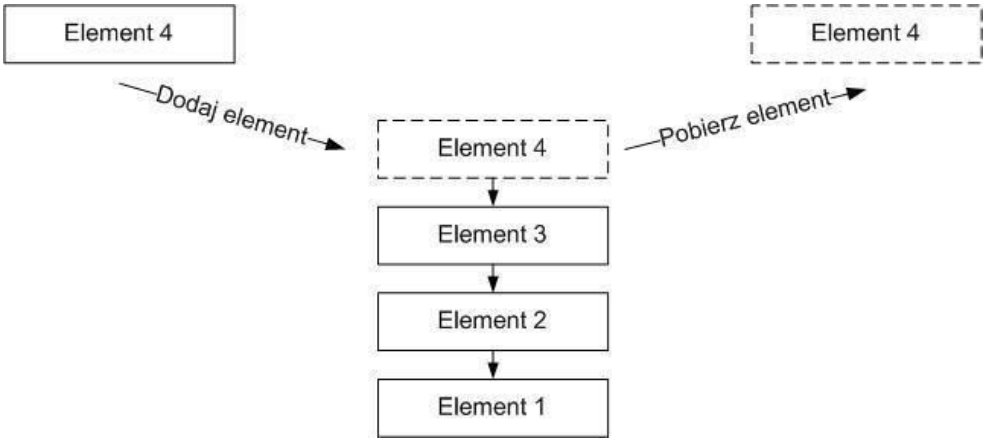
Jest kolekcją składającą się z liniowo uporządkowanych elementów. Charakterystyczną cechą kolejki jest to, iż nowe elementy dodawane do kolejki są dodawane na jej końcu, natomiast elementy, które są pobierane z kolejki, są pobierane z jej początku. Oznacza to, że element, który pierwszy został umieszczony w kolejce, pierwszy ją opuści. Kolejki zwane są także kolekcjami FIFO (ang. First In First Out).



Rys. 3 - Kolejka

4. Stos

Jest kolekcją składającą się z liniowo uporządkowanych elementów. Charakterystyczną cechą stosu jest to, iż użytkownik posiada dostęp jedynie do elementu znajdującego się na wierzchołku stosu, a nowy element stosu można dodawać jedynie na jego wierzchołek. Oznacza to, że element, który został umieszczony na stosie jako ostatni, pierwszy opuści stos. Stosy zwane są również kolekcjami LIFO (ang. Last In First Out).



Rys. 4 - Stos

Zadanie 1. Proszę zrealizować aplikację obiektową zgodnie z poniższymi założeniami:

- Należy zaimplementować własną kolekcję będącą kolejką.
- Elementy kolejki mają przechowywać dane typu „object”.
- Kolejka ma posiadać właściwość dostępową informującą o ilości elementów w kolejce.
- Kolejka ma udostępniać metodę „Wypisz”, wypisującą wszystkie wartości elementów kolejki.
- Sposób działania kolekcji należy przetestować za pomocą zamieszczonego poniżej kodu testowego.

```
Kolejka k = new Kolejka();
k.Dodaj(1);
k.Dodaj(5);
k.Dodaj(3);
k.Dodaj(8);
k.Wypisz();
Console.WriteLine("Liczba elementów: {0}", k.LiczbaElementow);
int element = (int)k.Pobierz();
Console.WriteLine("Pobrany element: {0}", element);
k.Wypisz();
Console.WriteLine("Liczba elementów: {0}", k.LiczbaElementow);
k.Dodaj(7);
k.Dodaj(4);
k.Wypisz();
Console.WriteLine("Liczba elementów: {0}", k.LiczbaElementow);
element = (int)k.Pobierz();
Console.WriteLine("Pobrany element: {0}", element);
k.Wypisz();
Console.WriteLine("Liczba elementów: {0}", k.LiczbaElementow);
Console.ReadKey();
```

Wskazówki dotyczące realizacji zadania:

- Należy zaimplementować klasę „Element”, zawierającą konstruktor oraz pola:
  - „wartość” typu „object”
  - „nastepnyElement” typu „Element” – pole to stanowi referencję do kolejnego elementu kolekcji
- Należy zaimplementować klasę „Kolejka”, zawierającą pola:
  - „pierwszyElement” typu „Element” - pole to stanowi referencję do pierwszego elementu kolekcji
  - „ostatniElement” typu „Element” - pole to stanowi referencję do ostatniego elementu kolekcji
  - „liczbaElementow” typu „int” - pole to ma zawierać informację o liczbie elementów znajdujących się w liście ▪ W klasie „Kolejka” należy zaimplementować metodę „Dodaj”, która ma dodawać element na koniec kolejki. Metoda sprawdza, czy w kolejce znajdują się elementy, odczytując wartość pola „pierwszyElement”. Jeśli ta wynosi „null”, wstawiany element będzie pierwszym i zarazem ostatnim elementem kolejki. Jeśli w kolejce znajdują się elementy, wstawiany element musi być ostatnim elementem kolejki – należy ustawić referencję „nastepnyElement” ostatniego elementu kolejki, aby odnosiła się do nowo

dodawanego elementu, oraz należy zmienić referencję „ostatniElement" klasy typu „Kolejka".

- W klasie „Kolejka" należy zaimplementować metodę „Pobierz", która ma usuwać pierwszy element kolejki i zwracać jego wartość. Metoda sprawdza, czy w kolejce znajdują się elementy, odczytując wartość pola „pierwszyElement". Jeśli ta wynosi „null", wyświetlony ma zostać odpowiedni komunikat i zwrócona ma zostać wartość „null". Jeśli w kolejce znajdują się elementy, należy pobrać jej pierwszy element i na podstawie zawieranej przez niego referencji „nastepnyElement" ustawić nowy pierwszy element klasy „Kolejka". Następnie należy zwrócić wartość pobranego elementu.

- Należy zwrócić uwagę, że wywołanie metod „Dodaj" i „Pobierz" może mieć wpływ na wartość pola „liczbaElementow". ■ W klasie „Kolejka" należy zaimplementować metodę „Wypisz", wypisującą wartości elementów kolejki. Wypisanie wartości elementu kolejki ma zostać zrealizowane za pomocą konstrukcji „Console.Write(wartosc.ToString())".

Zadanie 2. Proszę zrealizować aplikację obiektową zgodnie z poniższymi założeniami:

- Należy zaimplementować własną kolekcję będącą listą.

- Elementy kolejki mają przechowywać dane typu „Osoba".

- Klasa „Osoba" ma posiadać dwa pola – „imie" i „nazwisko", konstruktor przyjmujący parametry, oraz metodę „WypiszInfo" wypisującą informacje na temat imienia i nazwiska danej osoby.

- Lista ma zawierać metody umożliwiające: dodanie elementu na koniec listy, pobranie elementu listy o danym indeksie, wstawienie elementu do listy w miejsce o podanym indeksie.

- Lista ma udostępniać metodę „Wypisz", wypisującą wszystkie wartości elementów listy.

Sposób działania kolekcji należy przetestować za pomocą zamieszczonego poniżej kodu testowego.

```
Osoba o = new Osoba("Alicja", "Nowak");
Osoba o2 = new Osoba("Karolina", "Kowalska");
Osoba o3 = new Osoba("Michał", "Jabłoński");
Osoba o4 = new Osoba("Karol", "Wiśniewski");

Lista l = new Lista();

l.Dodaj(o);
l.Dodaj(o2);
l.Dodaj(o3);
l.Dodaj(o4);

l.Wypisz();

l.Pobierz(2);
l.Pobierz(0);
l.Pobierz(1);

l.Wypisz();

l.Wstaw(o3, 0);
l.Wstaw(o4, 1);
l.Wstaw(o, 2);

l.Wypisz();
```

Wskazówki dotyczące realizacji zadania:

- Należy zaimplementować klasę „Osoba".

- Należy zaimplementować klasę „Element", zawierającą konstruktor oraz pola:
  - „wartość" typu „Osoba"
  - „nastepnyElement" typu „Element" – pole to stanowi referencję do kolejnego elementu kolekcji

- Należy zaimplementować klasę „Lista", zawierającą pola:
  - „pierwszyElement" typu „Element" - pole to stanowi referencję do pierwszego elementu kolekcji
  - „liczbaElementow" typu „int" - pole to ma zawierać informację o liczbie elementów znajdujących się w liście

- W klasie „Lista" należy zaimplementować metodę „Dodaj", która ma dodawać element na koniec listy. Metoda sprawdza, czy w liście znajdują się elementy, odczytując wartość pola „pierwszyElement". Jeśli ta wynosi „null", wstawiany element będzie pierwszym elementem listy. Jeśli w liście znajdują się elementy, wstawiany element musi być ostatnim elementem listy – należy ustawić referencję „nastepnyElement" ostatniego elementu listy na nowo dodawany element. Ostatni element listy należy znaleźć za pomocą pętli „while".

- W klasie „Lista" należy zaimplementować metodę „Pobierz", przyjmującą jako parametr indeks, która ma usuwać z listy element o podanym indeksie i zwracać jego wartość. Metoda musi sprawdzać, czy podany indeks mieści się w wymaganym przedziale <0, liczbaElementowListy).

- W klasie „Lista" należy zaimplementować metodę „Dodaj", przyjmującą jako parametr indeks, która ma wstawiać do listy element w miejsce podanym indeksie. Metoda musi sprawdzać, czy podany indeks mieści się w wymaganym przedziale <0, liczbaElementowListy>.

- W klasie „Lista" należy zaimplementować metodę „Wypisz", wypisującą informacje na temat wartości elementów listy (imię i nazwisko osoby).

Zadanie do domu.

1. Proszę utworzyć własną kolekcję będącą stosem.

2. Proszę utworzyć własną kolekcję będącą listą dwukierunkową.

Zagadnienia, które należy uznać za przyswojone w trakcie zajęć. Po zajęciach będzie obowiązywać praktyczna znajomość:

- Pojęcie kolekcji.

- Pojęcie listy, stosu, kolejki.

- Zasady działania poznanych kolekcji.