מבוא לתכנות II

הרצאה 3 – דוגמאות נוספות למחלקות

סמסטר 2

כל דבר הוא מחלקה

- כל קטע קוד של פקודות נמצא בתוך פונקציה
- כל פונקציה נמצאת בתוך מחלקה
- מחלקה היא ההגדרה עבור אובייקט
- כל מה שנקודד יהיה מחלקות
- מופע הוא אובייקט של מחלקה
- מחלקה יכולה להכיל שדות = משתנים ומתודות = פונקציות

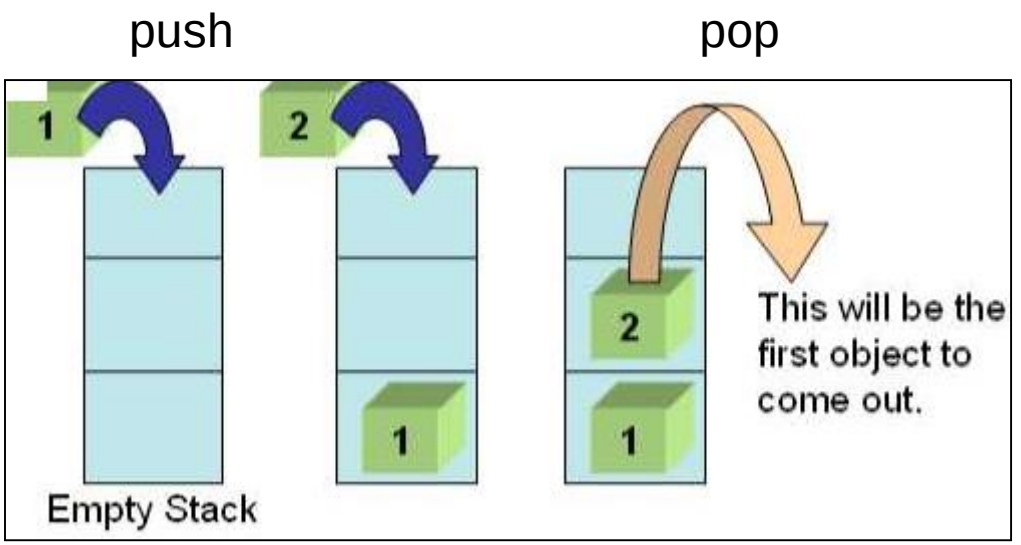
תרגיל

- ממשו את מחלקת "סטודנט" שלפניכם:

```
class Student
{
     //Fields
    private int Sid ;
    private int Sgrade ;
    private string Sname ;

    public void SetID(int newId )...

    public void  SetGrade(int newGrade )...

    public void SetName(string newName )...

    public void PrintStudent()...

}
```

מחסנית

- מבנה נתונים המממש את שיטת אחסון נאי"ר (LIFO)
- LIFO – Last In, First Out
- נַכנס אַחרון יַוצא רַאשון



מחסנית

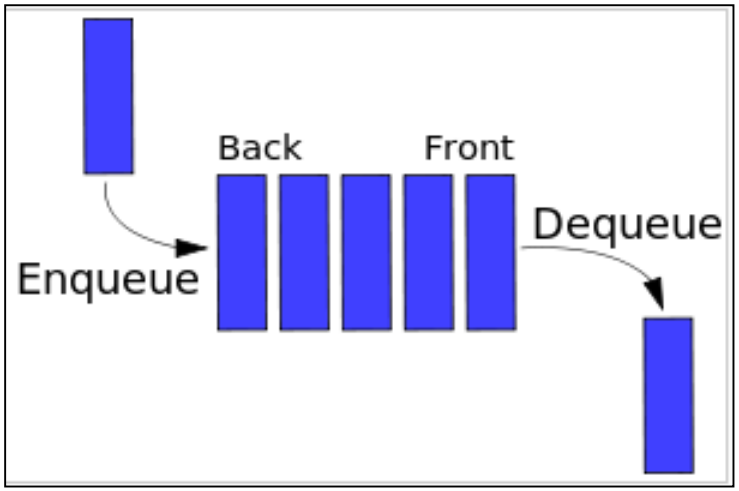
```
class Stack
{
    int[] values;
    int pointer; // points to the next available cell in the array

    // Default constructor
    public Stack()...

    // Constructor that gets the size of the stack!
    public Stack(int size)...

    //add an element to the stack
    public void Push(int newValue)...

    //remove an element from the stack
    public int Pop()...

    //returns the most (recent=) top element in the stack
    public int Peek()...

    //true ifthe stack is empty
    public bool IsEmpty()...
}
```

```
Stack s = new Stack(3);
s.Push(1);
s.Push(2);
s.Push(3);
s.Push(4);
s.Push(4);
Console.WriteLine(s.Pop());
Console.WriteLine(s.Pop());
Console.WriteLine(s.Pop());
Console.WriteLine(s.Pop());
Console.WriteLine(s.Pop());

Console.WriteLine("_____________________");
s.Push(1);
s.Push(2);
s.Push(3);
s.Push(4);
s.Push(4);

while (s.Peek() != -1)
{
    Console.WriteLine(s.Pop());
}
```

```
Stack is full!
Stack is full!
3
2
1
pop() - Stack is empty!
-1
pop() - Stack is empty!
-1
_____________________
Stack is full!
Stack is full!
3
2
1
peek() - Stack is empty!
Press any key to continue
```

תור

- מבנה נתונים המממש את שיטת אחסון נרי"ר (FIFO)
- FIFO – First In, First Out
- נַכנס רַאשון יַוצא רַאשון



תור

```
class Queue
{
    int[] arr;
    // Points to the empty cell
    int empty_cell = 0;

    // Default ctor - size 10!
    public Queue()...

    // Constructor that gets the size of the Queue
    public Queue(int size)...

    //add an element
    public void Enqueue(int value)...

    //remove an element
    public int Dequeue()...

    public bool IsEmpty()...

    public bool IsFull()...

    public void PrintQueue()...
}
```

```
Queue q2 = new Queue(3);
q2.PrintQueue();
q2.Enqueue(1);
q2.Enqueue(2);
q2.Enqueue(3);
q2.Enqueue(4);
q2.PrintQueue();
//q2.Enqueue(2);
//q2.PrintQueue();
Console.WriteLine(q2.Dequeue());
//q2.Enqueue(3);
//q2.Enqueue(4);
q2.PrintQueue();
Console.WriteLine("____________________");
Console.WriteLine(q2.Dequeue());
Console.WriteLine(q2.Dequeue());
q2.Enqueue(5);
Console.WriteLine(q2.Dequeue());
Console.WriteLine(q2.Dequeue());
```

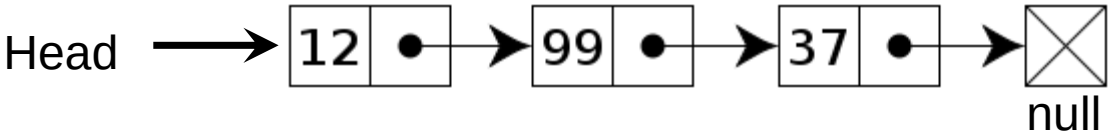
```
the queue:
Queue is full!
the queue: 1 2 3
1
the queue: 2 3
____________________
2
3
5
Queue is empty!
-1
Press any key to continue
```

תור

- מימוש תור יעיל יותר?

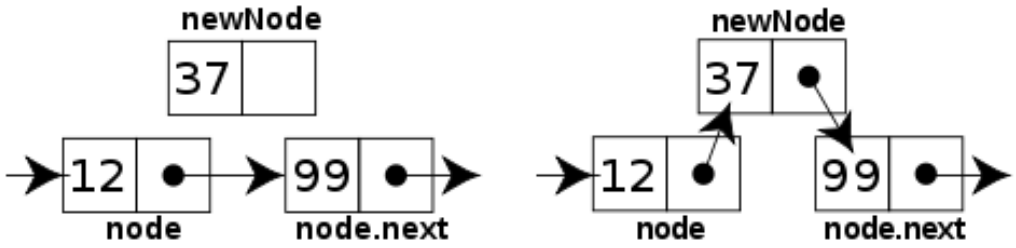
רשימה מקושרת

- מבנה נתונים שמייצג רשימה של קדקודים (nodes), כאשר כל קדקוד היא שילוב של מידע וקישור לקדקוד הבא.

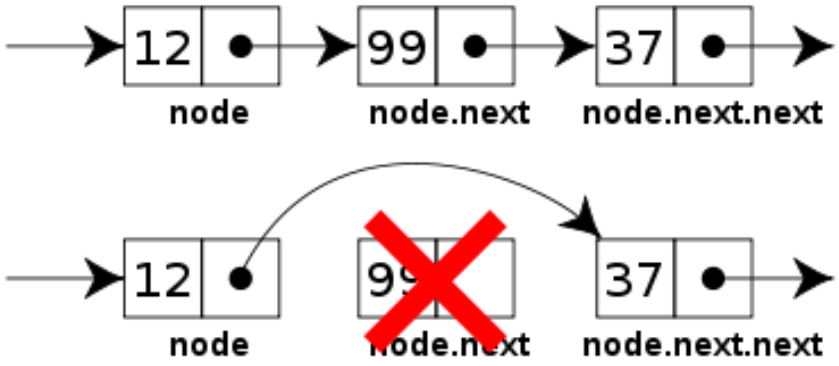


רשימה מקושרת

- הוספת קדקוד חדש לרשימה:



- הסרת קדקוד מהרשימה:



רשימה מקושרת

```
class LinkedList
{
    Node head;

    public LinkedList(int value)...

    public void InsertLast(int value)...

    public void InsertFirst(int value)...

    public void InsertAfter(Node temp, int value)...

    public void AddSorted(int valueToAdd)...

    public void DeleteList()...

    public bool Exists(int valueToFind)...

    public Node FindNode(int valueToFind)...

    public void RemoveFirst()...

    public void Remove(int valueToRemove)...

    public void PrintList()...
}
```

```
class Node
{
    public int value;
    public Node next;

    public Node(int value)
    {
        this.value = value;
    }
}
```

```
Node n1 = new Node(7);
Console.WriteLine(n1.Value);
Console.WriteLine(n1.Next==null);

LinkedList list3 = new LinkedList(1);
list3.InsertFirst(2);


LinkedList list = new LinkedList(28);
list.InsertLast(3); // second node in the list!
list.InsertLast(10);
list.InsertLast(4);
list.InsertFirst(1);
list.InsertAfter(list.FindNode(10),11 );

//Console.WriteLine(list.Exists(4));

//Node foundIt = list.FindNode(4);
//if(foundIt != null)
//    Console.WriteLine(foundIt.Value);
//else
//    Console.WriteLine("Not found!");

list.PrintList();
list.RemoveFirst();
list.PrintList();

list.Remove(10);
list.PrintList();

Console.WriteLine("*****************");
LinkedList list2 = new LinkedList(8);
list2.AddSorted(1);
list2.AddSorted(10);
list2.AddSorted(9);
list2.AddSorted(19);

list2.PrintList();

list2.DeleteList();
list2.AddSorted(1);

list2.PrintList();
```

```
7
True
1 28 3 10 11 4
28 3 10 11 4
28 3 11 4
*****************
1 8 9 10 19
1
Press any key to cont
```

יעילות

- מהו מבנה הנתונים הטוב ביותר?