



Priority Queue

code
academy

Yusif Imamverdiyev

TABLE OF CONTENTS

01

Stack

03

Priority Queue

02

Queue

04

HashSet<T>

INTRODUCTION

Mainly this type data structure is as same as Queue and Stack. This type of data structures are one dimension. And in the Priority Queue we will have priority variable. In another word in this type of data structures the element which is adding collection will have priority level and it placed in the first place





01

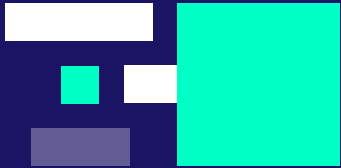
Stack



ABOUT Stack

Stack is one of the main data structures in the programming. Stack must have type and all elements have to be same type . Stack is working with LIFO logic. It means Last in first out. C# have generic Stack and non generic Stack collection classes. Let's stack functions and how it is working .

Main Methods of Stack



Push (T)

Insert variable front of Stack



Peek (T)

Return the first element

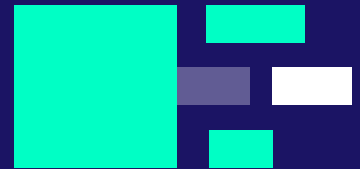


Pop()

Return the first element and remove it

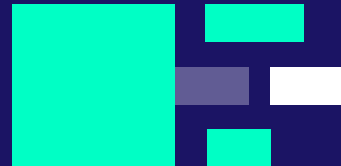
Contains(T)

Finds element in Stack.



Clear(T)

Delete all elements of stack



Let's use stack methods



Apple

Banana

Peach

Strawberries

Oranges

Push("Apple")

Let's use stack methods

Apple

Banana

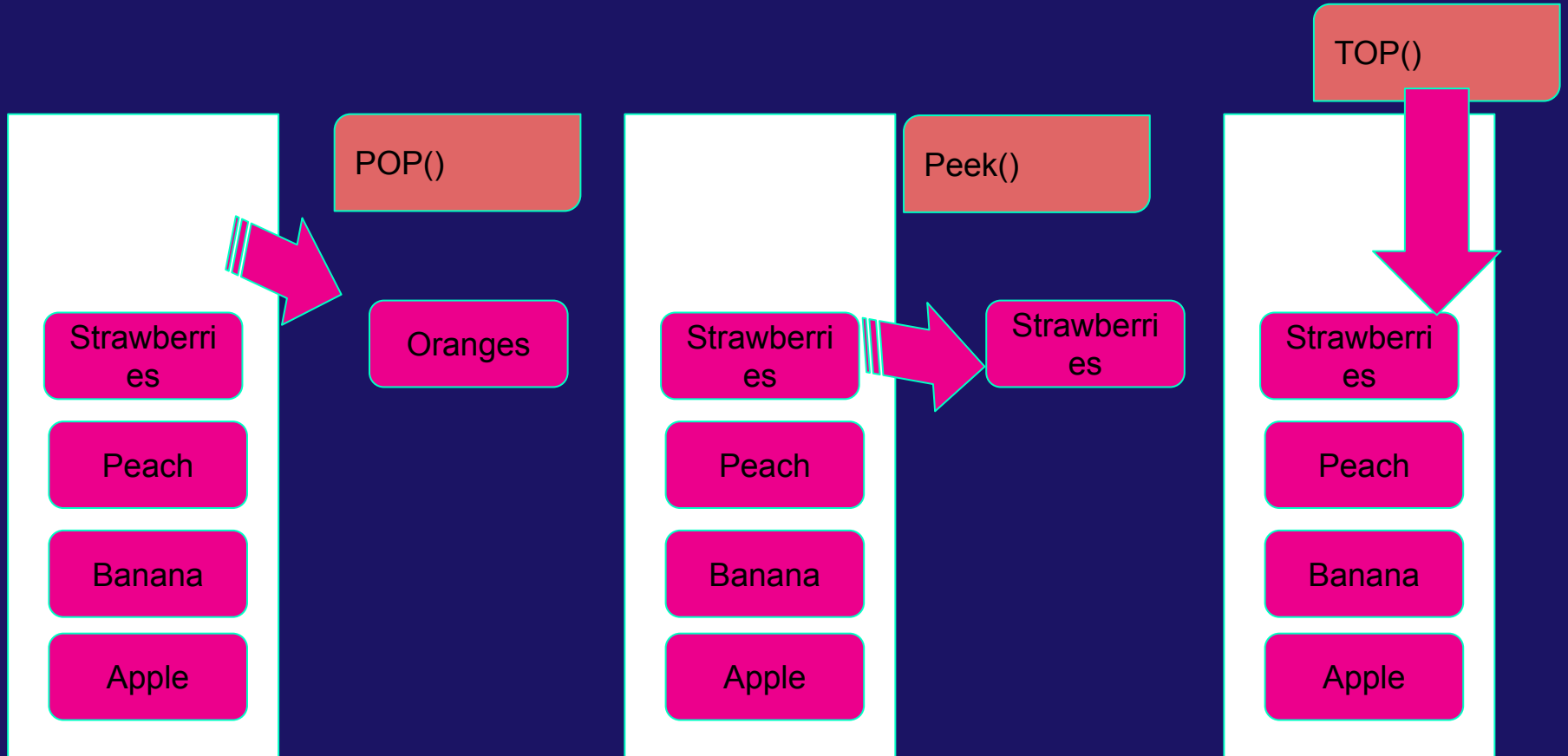
Peach

Strawberries

Oranges

```
Push("Banana")  
Push("Peach")  
Push("Strawberries")  
Push("Oranges")
```


Let's use stack methods



A photograph of a long queue of people waiting outside a light-colored brick building. The queue starts in the foreground on the left and extends towards the right, where it leads up a set of stairs to a doorway. The people are dressed in casual attire, including jackets, hoodies, and jeans. The scene is captured from a low angle, emphasizing the length of the line. Several colorful rectangular overlays (magenta, cyan, white, and dark blue) are present on the image, some partially obscuring the queue and others serving as a background for text.

02

Queue



ABOUT Queue

We have already take a look stack and common methods of stack. Now we will learn "Queue". Queue also most common data structure in programming. It is the opposite of Stack. It is FIFO style. First in First out .

Specs of Queue

Queue

Queue is a FIFO (First In First Out) collection.

It comes under the `System.Collection.Generic` namespace.

The queue can contain items of the specified type. It provides compile-time type checking and does not perform boxing-out of the box because it is generic.

Main Methods of Queue



Enqueue(T)

Insert variable front of Stack

Dequeue()

Return the first element and remove it



Peek (T)

Return the first element

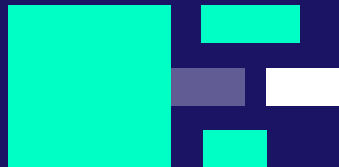
Contains(T)

Finds element in Stack.

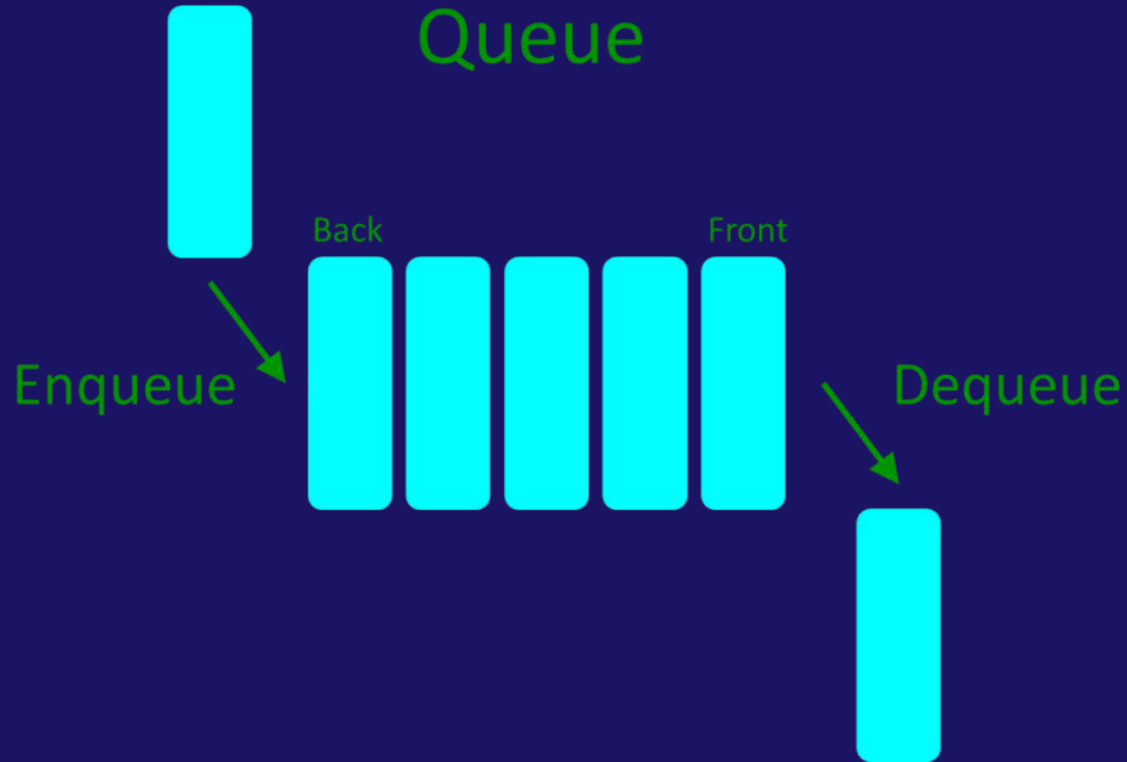


Clear(T)

Delete all elements of stack



Let's use queue methods



Let's use stack methods



Push("11")

Let's use stack methods

11

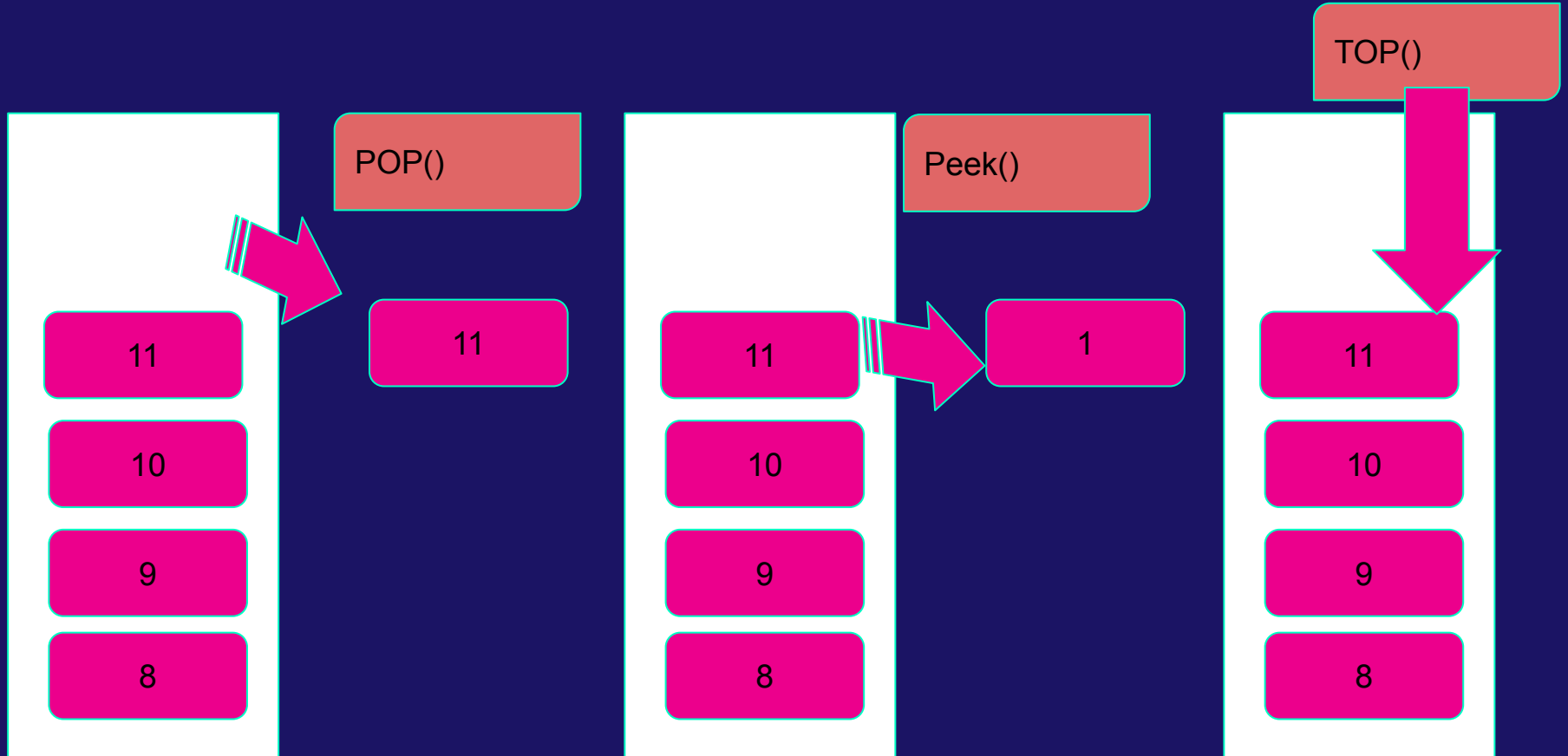
10

9

8

```
Push("10")  
Push("9")  
Push("8")
```


Let's use stack methods



03

Priority Queue



ABOUT Priority Queues

We have already take a look stack and queue. Now we can look Priority Queue. It is as similar as Stack and Queue also it has priority level. It comes with .Net 6 . This data structure come from generic namespace `als`.



Why ?

Up to present inside of .Net have types which are works FIFO (`Queue<T>`) and LIFO (`Stack<T>`) logic. Unlike this types `PriorityQueue<TElement,TPriority>` give opportunity to sort the items according another priority , different from arrival time. Using this type , we can set up “producer-consumer” structures with special priority logic

Specs of Queue

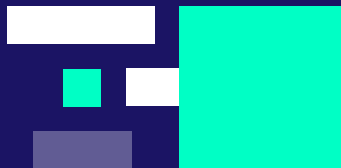
Priority Queue

- It takes two different generic parameter

- One of them is type of items and other is priority level type

- Priority queue uses quaternary min heap in internal implementation

Main Methods of Queue



Enqueue(T)

Insert variable front of Stack

Dequeue()

Return the first element and remove it

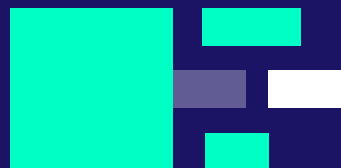


Peek (T)

Return the first element

Contains(T)

Finds element in Stack.



Clear(T)

Delete all elements of stack



```
//Creating Priority Queue
```

```
PriorityQueue<string, int> priorityQueue = new PriorityQueue<string, int>();
```

```
//Adding Items to Queue
```

```
priorityQueue.Enqueue("Apple", 4);
```

```
priorityQueue.Enqueue("Banana", 3);
```

```
priorityQueue.Enqueue("Orange", 1);
```

```
priorityQueue.Enqueue("Strawberry", 5);
```

```
priorityQueue.Enqueue("Grape", 0);
```

```
priorityQueue.Enqueue("Pineapple", 6);
```

```
priorityQueue.Enqueue("Peach", 7);
```

Item

Priority

Apple

4

Peach

7

Grape

0

Orange

1



Item

Priority

Grape

0

Orange

1

Apple

4

Peach

7

Fruit Name : Grape Priority : 0

Fruit Name : Orange Priority : 1

Fruit Name : Banana Priority : 3

Fruit Name : Apple Priority : 4

Fruit Name : Strawberry Priority : 5

Fruit Name : Pineapple Priority : 6

Fruit Name : Peach Priority : 7

----- After using Dequeue -----

Priority Queue size = 0

C:\Users\yusif\source\repos\PriorityQueue\PriorityQueue\bin\Debug\net6.0\PriorityQueue.exe (process 45880) exited with code 0.

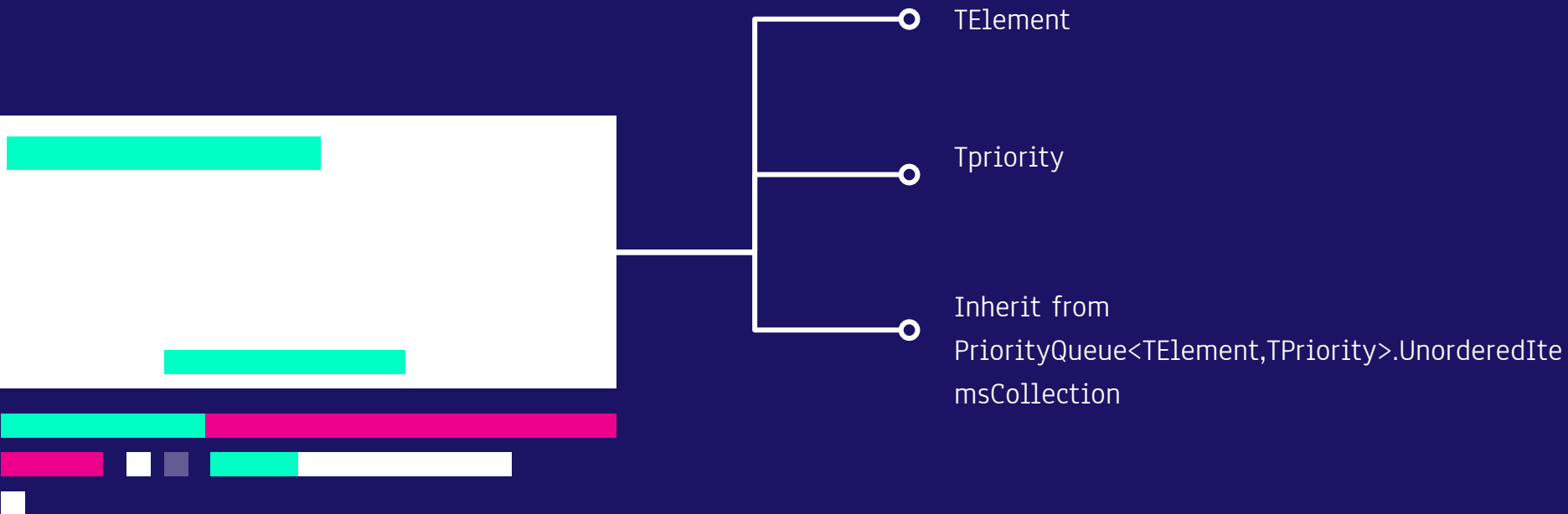
Press any key to close this window . . .

`PriorityQueue<TElement,TPri
ority>.UnorderedItems Collec
tion Class`

About content

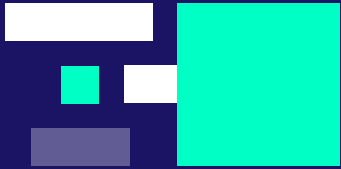
Enumerates the contents of a
`PriorityQueue<TElement,TPriority>`, without
any ordering guarantees

Specs



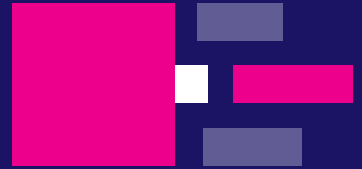
HashSet<T>

Main Methods of HashSet



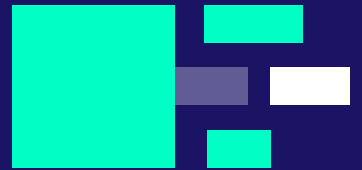
`Equals(Object)`

`GetType()`



`GetEnumerator()`

`GetHashCode()`



THANKS !

Do you have any questions?
yusif.pi@code.edu.az

code
academy