

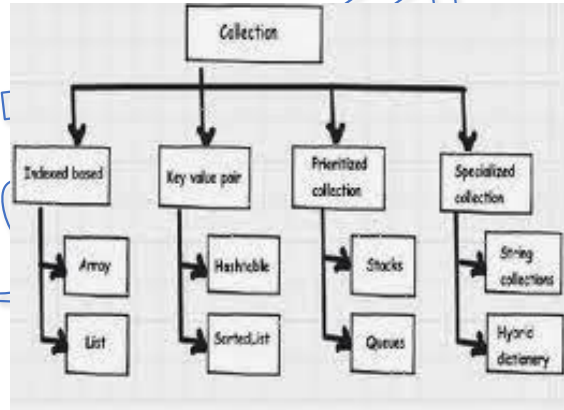
برمجة - ٣ -

Collections part-1

المحاضرة الخامسة

Collections

- التجميعات هي عبارة عن فئات خاصة تستخدم لتخزين واستعادة البيانات تستخدم للوصول الى مواقع الذاكرة الديناميكية وغيرها من الاستخدامات أنواع فئات المجموعات أو التجميعات واستخدامها
- جميع التجميعات تستخدم فضاء الاسماء . `system.collections`



تستخدم المفتاح مثلما يستخدم الفهرس للوصول لعناصر القائمة القائمة المرتبة هي عبارة عن تركيب أو خليط من المصفوفات وجداول المجزئة.

تحتوي على قائمة من العناصر التي يمكن الوصول اليها بالمفتاح أو الفهرس :

إذا أردت الوصول للعناصر بواسطة الفهرس فانه يستخدم مصفوفات القوائم وفي حال استخدم المفتاح للوصول للعناصر فانه يستخدم جدول التجزئة

. دائما يتم ترتيب العناصر بواسطة قيمة المفتاح

Stack

انها تمثل الداخل أخيرا خارج أولا لكائن المجموعة ويعرف بالمكدس
تستخدم عندما نحتاج للوصول لعناصر نريدها
الداخل أخيرا خارج أولا. وعندما نحتاج
إضافة عنصر فانها تسمى
PUSHING
وعندما نريد حذف عنصر تسمى popping

Queue

الرتل ويمثل الداخل أولا خارج أخيرا لكائن المجموعة
تستخدم عندما نحتاج للوصول لعناصر الداخل
أولا خارج أخيرا . عندما تستخدم لإضافة
عنصر في القائمة فانها تستدعي
enqueue
وعندما تريد حذف عنصر تستدعي
deque

ArrayList

تمثل لمجموعة مرتبة لكان يمكن فهرسته فردا فرد وهي في الأساس بدي ل للمصفوفات ومع ذلك في لا تشبه المصفوفات حيث يمكنك اضافة وحذف العناصر من القائمة في الموقع المحدد باستخدام الفهرس وحيث المصفوفة تقوم بتغير حجم المصفوفة بنفسها تلقائيا. كما أنها تسمح بتخصيص المساحة في الذاكرة الديناميكية وتضيف وتبحث وترتيب العناصر في القائمة.

Hashtable

تستخدم مفتاح للوصول للعناصر في المجموعات الجدول المجزء يستخدم عندما نحتاج للوصول للعناصر باستخدام المفتاح وبامكانك تعريف قيمة مفتاحية تستخدمها. وكل عنصر في الجدول المجزء له زوج من المفتاح وقيمه. حيث المفتاح يستخدم للوصول للمجموعة

BitArray

مصفوفة البت هي مصفوفة تمثل بالنظام الثنائي أما بقيمة ٠ أو ١
تستخدم عندما تريد تخزين البتات وانت لا تعرف عدد البتات عموماً. يمكنك الوصول للعناصر مصفوفة البتات للمجموعة باستخدام الفهرس العدد الصحيح ويبدأ من الصفر

المكذسات

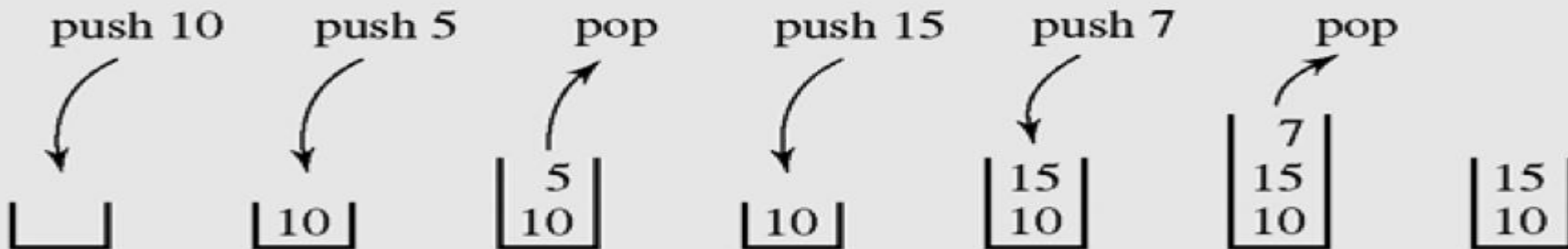
م. تغريد حرفوش

المكدس STACK CLASS

المكدس: يمثل طريقة حفظ البيانات والوصول اليها في الذاكرة حسب

"الداخل أخيرا خارج أولا"

عندما يتم اضافة عنصر للمكدس يسمى pushing
وعند الحذف من المكدس يسمى popping .



المكدسات هي:

- بعبارة أخرى هي عبارة عن خط انتظار لمجموعة من البيانات، ما يميز هذا الخط بأنه مفتوح من اتجاه واحد فقط
- أي أن البيانات تدخل وتخرج من بوابة واحدة. يطلق على stack وLIFO وهذا الرمز يعني بأن
- آخر عنصر دخل إلى stack هو أول عنصر يغادرها،
- وهذا بديهي بما أنه لا يوجد لدينا إلى فتحة واحدة لإدخال وإخراج العناصر.
- stack يمكننا أن نشبهها بحاملة الأقراص المدمجة فأخر قرص تقوم بوضعه هو أول قرص تقوم بإخراجه، أو كمجموعة
- كتب متراسة فوق بعضها البعض:
-



لماذا نستخدم stack؟

- يمكننا استخدام stack في عدة مواقف هذه بعض منها:
- - في حالة وجود بيانات ونريد عرضهم بالمعكوس، فعندها ندفع هذه العناصر الى stack وعند إخراجهم فإن آخر عنصر سوف يخرج أولاً.
- - نستخدم في لغات البرمجة للتحقق من وجود تطابق بين الرموز المدخلة (مثلاً للتأكد بأن كل { لديها }).
- - نستخدم لإضافة الأعداد الكبيرة باستخدام بعض الخوارزميات.

التسلسل	دوال المكس	الاستخدام
1	void Clear();	ازالة كل العناصر من المكس
2	bool Contains(object obj);	تحديد إذا ماكان العنصر في المكس
3	object Peek();	تعيد قيمة العنصر الموجود في أعلى المكس بدون حذف
4	object Pop();	تعيد قيمة العنصر الموجود في أعلى المكس مع ازالته من المكس
5	void Push(object obj);	اضافة عنصر الى اعلى المكس
6	object[] ToArray();	نسخ المكس الى مصفوفة
التسلسل	الخاصية	الاستخدام
	Count	قراءة عدد العناصر التي يحتويها المكس

مثال:

```
using System;
using System.Collections;
namespace CollectionsApplication {
class Program {
static void Main(string[] args) {
    Stack st = new Stack(); st.Push('A');
    st.Push('M'); st.Push('G'); st.Push('W');
    Console.WriteLine("Current stack: ");
    foreach (char c in st)
    {
        Console.Write(c + " ");
    }
    Console.WriteLine();
    st.Push('V');
    st.Push('H');
    Console.WriteLine("The next poppable value in stack: {0}", st.Peek());
    Console.WriteLine("Current stack: ");
    foreach (char c in st)
    { Console.Write(c + " "); }
    Console.WriteLine();
    Console.WriteLine("Removing values ");
    st.Pop();
    st.Pop();
    st.Pop();
    Console.WriteLine("Current stack: ");
    foreach (char c in st) { Console.Write(c + " "); }
    } } }
```

ناتج المثال:

Current stack:

W G M A

The next poppable value in stack: H

Current stack:

H V W G M A

Removing values

Current stack:

G M A

م. تغريد حرفوش

مثال عن الفرز:

```
public static void main(String[] args) {  
    Stack myStack = new Stack();  
    int[] a = {1, 2, 3, 4, 5, 6, 7, 8, 9};  
    Console.WriteLine("WE WANTE TO SORT THIS VALUES");  
    for(int i =0; i < 9; i++){  
        myStack.push(a[i]);  
        Console.WriteLine(myStack.topEl() + " ");  
    }  
  
    Console.WriteLine("THE VALUES AFTER WE PUT THEM IN STACK+\n");  
    for(int i = 0; i < 9; i++){  
        Console.WriteLine(myStack.pop() + " ");  
    }  
}
```

في البداية قمنا بإضافة عناصر المصفوفة الى Stack وكذلك
وفي نفس loop قمنا بطباعة
أعلى عنصر موجود وهو آخر عنصر قمنا بإضافته
ومن ثم قمنا بإخراج العناصر من stack واحد تلو الآخر مع طباعتهم.

فما هو الخرج النهائي ???!!

الارتال

Collections part-2-

المحاضرة السادسة



الرتل

QUEUE CLASS

- تستخدم عندما تريد حفظ بيانات بحيث يكون
- "الداخل أولا خارجا أولا"
- عند الحذف من الرتل تستخدم الدالة **deque**
- عند الاضافة الرتل تستخدم الدالة **enqueue**

حرفوش

Queue

- الرتل بمعنى اخر هو اشبه بطابور انتظار وهذا الطابور يزداد حجمه بإضافة البيانات في آخره ويقل بإخراج البيانات من الأمام.
- المعنى العام شبيه بالطوابير العادية حيث أن من يأتي أولاً يخرج أولاً والمتأخر يقف في آخر الصف
- إذا على عكس stack فإن Queues لديه نهاية ومقدمة
- النهاية لدخول البيانات والمقدمة لخروجها.
- كتشبيه عنه طابعة الورق ...

حرف فوش

أين نستخدم Queue

- نستخدم في تطبيقات خدمة العملاء فعندها نضيف آخر واصل في آخر الرتل ونقوم بخدمة الأول واخراجه من الرتل.
- نستخدم في ترتيب العمليات في CPU بعض الأحيان يستعان بـ Queue تقدم المهم أولاً.
- الاستعلام عن العناصر في Tree وهي عبارة عن طريقة لحفظ البيانات كما القوائم والمصفوفات.

حرفوش

التسلسل	دوال الرتل	الاستخدام
1	void Clear();	ازالة كل العناصر من الرتل
2	bool Contains(object obj);	تحديد إذا ما كان العنصر في الرتل
3	object Dequeue();	ازالة وارجاع العنصر من بداية الرتل
4	void Enqueue(object obj);	اضافة عنصر الى نهاية الرتل
5	void TrimToSize();	كتابة سعة العناصر الفعلية في الرتل
6	object[] ToArray();	نسخ الرتل الى مصفوفة
التسلسل	الخاصية	الاستخدام
1	Count	قراءة عدد العناصر التي يحتويها الرتل

```
using System;
using System.Collections;
namespace CollectionsApplication
{
    class Program
    {
        static void Main(string[] args)
        {
            Queue q = new Queue();
            q.Enqueue('A');
            q.Enqueue('M');
            q.Enqueue('G');
            q.Enqueue('W');
```

م. تغريد حرفوش

```
Console.WriteLine("Current queue: ");
    foreach (char c in q)
        Console.Write(c + " ");
    Console.WriteLine();
    q.Enqueue('V');
    q.Enqueue('H');
    Console.WriteLine("Current queue: ");
    foreach (char c in q)
        Console.Write(c + " ");
    Console.WriteLine();
    Console.WriteLine("Removing some values");
    char ch = (char)q.Dequeue();
    Console.WriteLine("The removed value: {0}", ch);
    ch = (char)q.Dequeue();
    Console.WriteLine("The removed value: {0}", ch);
    Console.ReadKey();
} }
```

Current queue:

A M G W

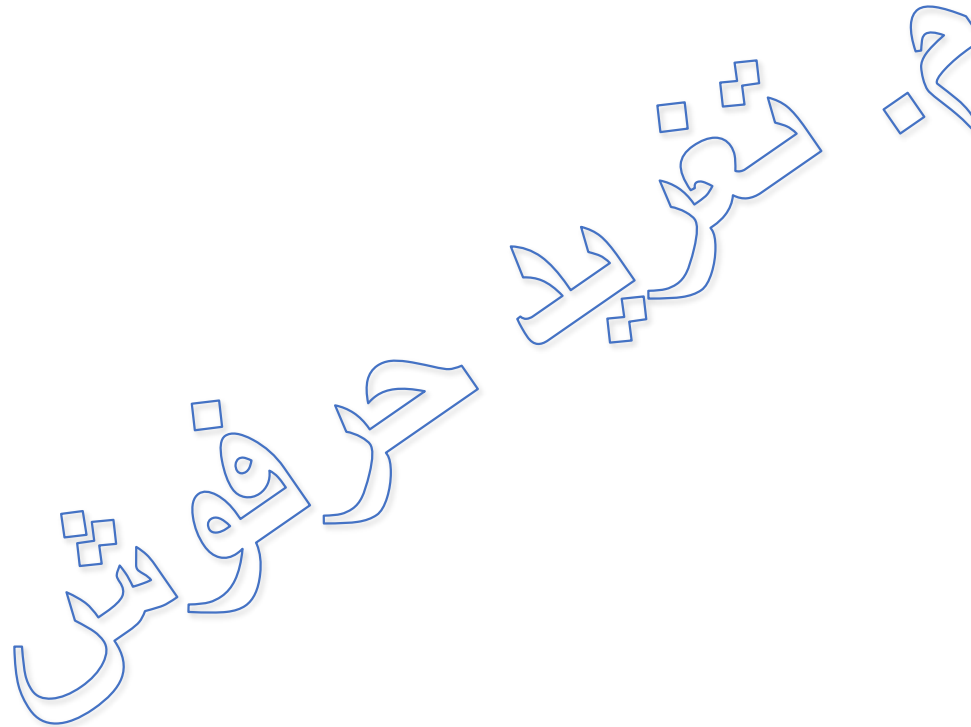
Current queue:

A M G W V H

Removing values

The removed value: A

The removed value: M



الخلاصة:

- المكدسات والارتنال من أنواع هياكل البيانات **Data Strucutre** يستخدمان لتخزين المعلومات في الذاكرة وكل نوع له اليه في التعامل مع البيانات
- Stack يستخدم LIFO Last In First Out
- Queue يستخدم FIFO First In First Out

حرفوش