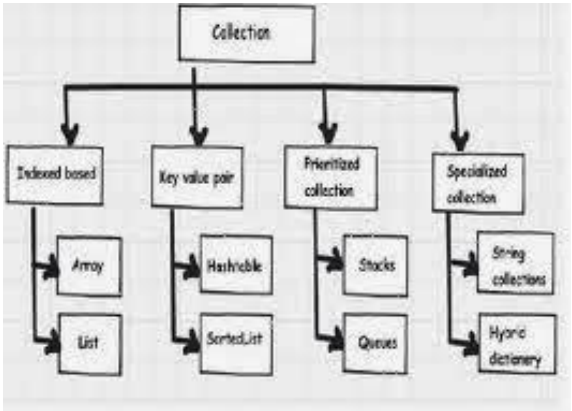
برمجة متقدمة - ١ -

Collections

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



النوع او الفئة	الوصف و الاستخدام
SortedList	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 أضافة 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 أخير . عندما تستخدم لاضافة عنصر في القائمة فانها تستدعي enqueue وعندما تريد حذف عنصر تستدعي deque

ArrayList	تمثل لمجموعة مرتبة لكان يمكن فهرسته فردا فرد وهي في الاساس 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 الذواكر الديناميكية وتضيف وتبحث وترتيب العناصر في القائمة.
Hashtable	تستخدم مفتاح للوصول للعناصر في المجموعات الجدول المجزء يستخدم عندما نجناتج للوصول للعناصر باستخدام المفتاح وبامكانك تعريف قيمة مفتاحية تستخدمها. وكل عنصر في الجدول المجزء له زوج من المفتاح وقيمته.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 عموما. يمكنك الوصول للعناصر مصفوفة البتات للمجموعة باستخدام الفهرس العدد الصحيح ويبدا من الصفر

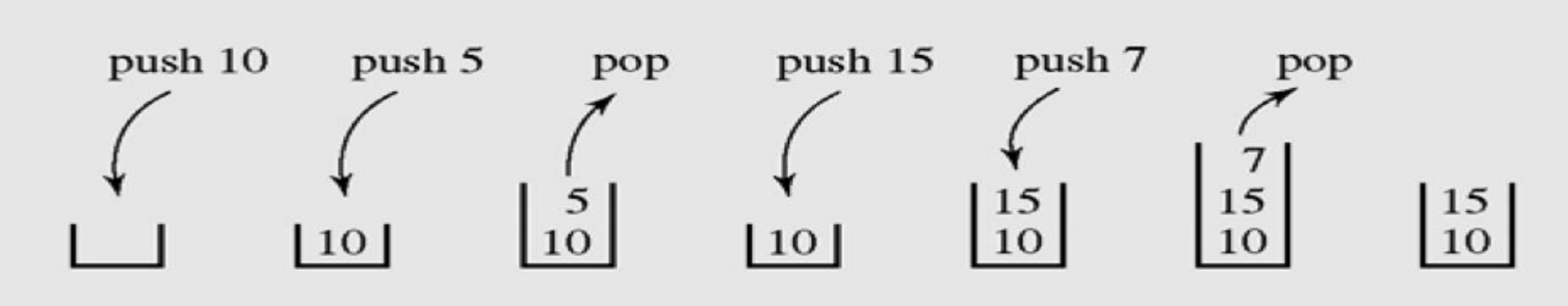
المكدسات

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 stackبـ LIFOوهذا الرمز يعني بأن
- آخر عنصردخل إلى stackهو أول عنصر يغادرها،
- وهذا بديهي بما أنه لا يوجد لدينا الى فتحة واحدة لإدخال وإخراج العناصر.
- stack يمكننا أن نشبهها بحاملة الأقراص المدمجة فآخر قرص تقوم بوضعه هو أول قرص تقوم بإخراجه، أو كمجموعة
- كتب متراصة فوق بعضها البعض:
-

لماذا نستخدم stack؟

- يمكننا استخدام stackفي 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.
- - تستخدم في لغات البرمجة لتتحقق من وجود تطابق بين الرموز المدخلة (مثلاً للتأكد بأن كل { لديها }).
- - تستخدم لإضافة الأعداد الكبيرة باستخدام بعض الخوارزميات.

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

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

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 باضافته
ومن ثم قمنا باخراج العناصر من stack واحد تلو الآخر مع طباعتهم.

فما هو الخرج النهائي ؟؟؟!!