الباب الثاني

خوارزمية الترتيب الانتقائي:

في الباب السابق ذكرنا مصطلحات مثل ال Array وال linked lists وفي هذا الباب سنتعلم الفارق بينهم، كما سنتعلم أول خوارزمية ترتيب لأن هناك بعض الخوارزميات التي لا تعمل إلا إذ كانت البيانات (data) مرتبة.

الذاكرة:

كيف تعمل الذاكرة؟ تخيل أنك تملك خزانة مليئة بالأدراج،
كل درج منهم يمكنك أن تخزن فيه شيء واحد فقط (هاتف / نظارة / قلم / حقيبة)
وكل منهم له رقم خاص يميزه عن غيره من الأدراج.
هذا ما يحدث في ذاكرة الكومبيوتر، الكومبيوتر يمتلك مجموعة كبيرة من الأدراج، كل منهم له عنوان مميز يمكنك استخدامه للوصول إلى القيمة المخزنة.

ولكن ماذا لو أردنا أن نخزن أكثر من قيمة؟ هنا يأتي دور ال Array وال Linked List.

ولكن ما الفارق؟ حين نستخدم ال Array نقو بتخزين القيم بشكل متتالي في الذاكرة، في هذا المثال سنخزن قائمة مشتريات

X	X	تفاح	طماطم	لبن
X				

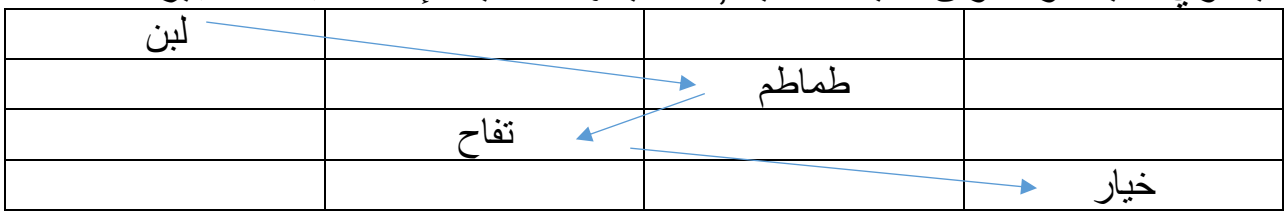
لو أردنا تخزين قيمة إضافية لن نستطيع تخزينها بجوارهم لأنها تحمل قيمة أخرى لذا سنضطر لنقل كل قيم ال Array إلى مكان يسع جميع القيم

X	X			
X	خيار	تفاح	طماطم	لبن

كلما أضفنا عنصر جديد نحتاج إلى البحث عن مجموعة متتالية من الأماكن الفارغة و الانتقال إليها، قد يبدو الأمر بسيطا، ولكن ماذا لو كنا نخزن ألف عنصر؟ يمكن تجنب هذه المشكلة بحجز أماكن من البداية، ولكن هذا قد يسبب مشاكل أخرى

- قد لا تحتاج إلى استخدام كل الأماكن المحجوزة مما يؤدي إلى إهدار الموارد.
- قد تحتاج إلى أماكن أكثر و ستضطر أيضا للانتقال.

ماذا عن ال Linked list؟ يمكننا التخزين في أي عنصر في الذاكرة، كل عنصر يحتوي القيمة و عنوان القيمة التالية , مما يسهل عملية الإضافة بشكل كبير



ما هي فائدة الArray؟ يمكننا الوصول إلى أي قيمة داخل ال array بسهولة لأن القيم متوالية فيمكننا الوصول إلى العنصر من خلال ال Index و هو عدد الخطوات داخل للوصول للعنصر المطلوب بداية من العنصر الأول

0->o	1->s	2->a	3->m	4->a

أيهما أفضل؟ الإجابة تعتمد على احتياجاتك , فإذا كان عدد التعديلات محدود و لكنك ستحتاج إلى استخدام القيم بكثرة فالأفضل هو ال array أما إذا كنت ستقوم بالعديد من التعديلات (الإضافة و الحذف) و استخدام القيم قليل فالأفضل هو ال linked list , المقارنة التالية للتوضيح

العملية	Array	Linked list
الإضافة (insertion)	تحتاج إلى نقل جميع القيم إلى مكان آخر مناسب مما يستهلك الوقت إذا لم يكن هناك مجموعة أماكن مناسبة تفشل عملية الإضافة O(n)	يمكن ببساطة إضافة أي عنصر بأي مكان بجعل العنصر السابق يخزن عنوانه (يشير إليه) إذا لم يكن هناك مكتن في الذاكرة تفشل عملية الإضافة O(1)
الحذف (deletion)	تحتاج لتحريك العناصر التالية للعنصر المحذوف لا يمكن أن تفشل O(n)	جعل العنصر السابق للمحذوف يشير للعنصر التالي له لا يمكن أن تفشل O(1)
استدعاء (قراءة و تعديل)	يمكن الوصول مباشرة من خلال ال index في O(1)	يجب المرور على كل العناصر السابقة أولا O(n)

ما هو الترتيب الانتقائي (selection sort)؟ لنفترض أن لديك مجموعة أرقام تريد أن ترابها تصاعديا {5,3,1,4,2} , ستقوم بالبحث عن أصغر رقم و تضيفه إلى array جديد ثم تبحث عن العنصر التالي و سيكون الترتيب هكذا

1- {5,3,4,2}, {1}

2- {5,3,4}, {1,2}

3- {5,4}, {1,2,3}

4- {5}, {1,2,3,4}

5- {}, {1,2,3,4,5}

سنجد أننا بحثنا في array حجمه n عدد n من المرات لذا سيكون n*n أي O(n^2)