شبكة الخوارزمي

مدخل سريع لخوارزميات البحث و الترتيب المختلفة

مقدمة:

المصفوفات Arrays و القوائم المرتبطة likned lists من بنى المعطيات الأساسية التي تُستخدم لتخزين المعلومات. و من العمليات الأساسية على بنى المعطيات هذه، البحث عن قيمة محددة من بين القيم المخزنة ضمن هذه البُنى.

سنتحدث فيما يلي عن الطرق المختلفة المستخدمة لإجراء عملية البحث هذه.

1- المصفوفات Arrays:

لنأخذ على سبيل المثال مصفوفة رقمية مكونة من سبعة عناصر. المشكلة المطروحة إيجاد عنصر محدد في هذه المصفوفة. أسهل طرق البحث تُدعى "البحث التسلسلي". و الموضحة بالخوارزمية التالية.

```
int function SequentialSearch ( Array A, int Lb, int Ub, int Key) ;
begin
    for i = Lb to Ub do
        if A[i] = Key then
            return i;
        returen -1; /*العنصر المطلوب غير موجود*/
end;
```

إذا كان العنصر المراد البحث عنه موجود في آخر المصفوفة أو إذا كان العنصر غير موجود في المصفوفة نحتاج في هذه الحالة إلى عدد من عمليات المقارنة(if A[i] = Key then) يساوي إلى عدد عناصر المصفوفة و هذه هي أسوء حالة ممكنة. أما الحالة الأحسن فتكون إذا كان العنصر المطلوب في أول المصفوفة حيث يتم العثور عليه بعد عملية مقارنة واحدة فقط. طبعاً عدد عمليات المقارنة يُعبر عن زمن أو سرعة تنفيذ هذه الخوارزمية.

إذا كانت عناصر المصفوفة مرتبة تسلسلياً فيمكن إستخدام خوارزمية البحث الثنائي و التي يمكن تلخيصها على الشكل التالي:

نبدأ بفحص العنصر الموجود في منتصف المصفوفة (أي العنصر ذو الترتيب M=Array length/2) فإذا كان العنصر

المطلوب إيجاده أصغر من العنصر الموجود في الوسط (و على إعتبار أن المصفوفة مرتبة تصاعدياً أي من العنصر الأصغر للعنصر الأكبر) فهذا يعني أن العنصر المطلوب موجود في النصف الأعلى من المصفوفة الأصلية. و بالتالي يتم البحث عن العنصر المطلوب في القسم العلوي من المصفوفة و ينقص عدد العناصر المطلوب فحصها بمقدار النصف.

يمكننا أن نتصور الآن أن المشكلة أصبحت إيجاد العنصر المطلوب في مصفوفة مرتبة تصاعدياً تُشكل مصفوفة جزئية من المصفوفة الأصلية (النصف العلوي أو السفلي فقط). و يتم تكرار نفس العملية السابقة على المصفوفة الجزئية الجديدة و هكذا حتى الوصول للعنصر المطلوب.

```
int function BinarySearch (Array A, int Lb, int Ub, int Key);
begin
    do forever
        M=(Lb+Ub)/2; /*تحديد ترتيب العنصر المتوسط */
        if (key< A[M]) then
            Ub = M-1; /* تحديد الحد الأعلى
            للمصفوفة الجديدة */
        else if (Key > A[M]) then
            Lb=M+1; /* تحديد الحد الأدنى للمصفوفة
            الجديدة */
        else
            return M; /*العنصر موجود */
        if (Lb > Ub) then
            return -1; /* العنصر غير موجود */
end;
```

0	4	← Lb
1	7	
2	16	
3	20	← M
4	37	
5	38	
6	43	← Ub

لنفترض أنه لدينا مصفوفة من 1023 عنصر بعد عملية المقارنة الأولى يتم تضييق نطاق البحث ليشمل 511 عنصر فقط و في عملية المقارنة الثانية يصبح لدينا 255 عنصر فقط و هكذا. و بالتالي نحتاج إلى عشر عمليات مقارنة فقط لفحص كامل المصفوفة و هذا أفضل بكثير من عملية البحث التسلسلي التي تحتاج في هذه الحالة إلى 1023 عملية مقارنة (لاحظ الفرق الكبير في عدد عمليات المقارنة) .

بالطبع عند استخدام خوارزمية البحث الثنائي تبرز مشكلة جديدة هي مشكلة ترتيب العناصر تصاعدياً أو تنازلياً. و عند حذف

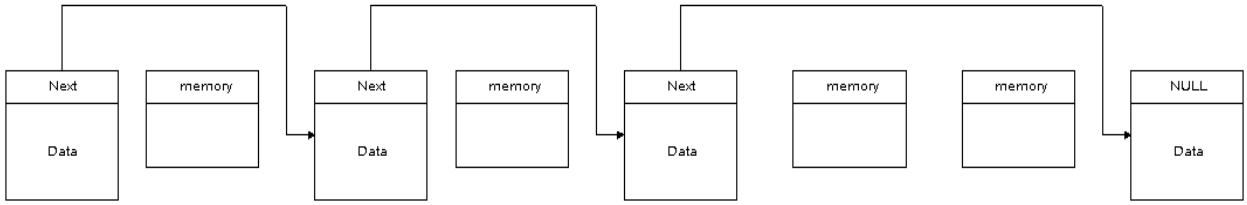
أو إضافة عنصر جديد لابد من إعادة ترتيب المصفوفة من جديد.

هنا تبرز القوائم المتسلسلة linked lists كبنى معطيات فعّالة تتمتع ببعض المرونة التي تسمح بحذف أو إضافة عناصر جديدة.

2- القوائم المرتبطة (Linked list)

على عكس المصفوفات فإننا لا نحتاج لمعرفة عدد العناصر المراد تخزينها في القوائم المرتبطة لأن هذه القوائم تمتاز بقدرتها على التوسع الغير محدود (و بشكل أدق التوسع بما تسمح به ذاكرة الحاسب المستخدم)

كل عنصر من القائمة يحوي قيمتين (و في القوائم الأكثر تعقيداً تستخدم ثلاث قيم) القيمة الأولى تمثل المعلومات المراد تخزينها و التي قد تكون بنية معقدة مثل السجل Record المستخدم في لغة باسكال او البنية Structure المستخدم في لغة C القيمة الثانية تحوي رقم الذاكرة المستخدمة في تخزين العنصر التالي هذه القيمة عادة تدعى مؤشر Pointer كما هو موضح في الشكل التالي:



من المهم ملاحظة أن العنصر الأخير في القائمة لا يشير إلى أي موقع و إنما يحوي القيمة NULL كمؤشر على إنتهاء القائمة

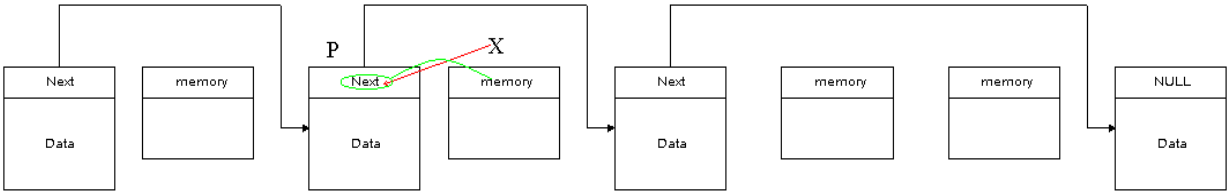
هناك العديد من المهام الأساسية التي يجب القيام بها على القوائم المرتبطة هذه العمليات تشمل:

أ- إضافة عنصر جديد

على افتراض اننا نريد اضافة عنصر مخزن في الموقع X بعد العنصر المخزن في الموقع P يتم ذلك كما يلي

```
X- > next = P - > next;

P - > next = X;
```



ب- حذف عنصر

سنترك هذه العملية كتدريب للأخ القارئ

ج- ترتيب القائمة

ترتيب القائمة يتم عن طريق اجراء مسح تسلسلي على عناصر القائمة حتى يتم العثور على الموقع المناسب و هذا بالطبع يتم على حساب الوقت

كما لاحظنا فإن خوارزميات البحث تعتمد على كون المعلومات مخزنة بشكل مرتب لذلك سنورد الآن بإيجاز الخوارزميات المستخدمة لترتيب المعطيات