مفاهيم أساسية في الخوارزميات وبنى المعطيات

المحاضرة الأولى
م. لمى السبع

مفاهيم هامة

- **الخوارزمية:**
- هي مجموعة من الخطوات الرياضية والمنطقية والمتسلسلة اللازمة لحل مشكلة ما بعدد منته من الخطوات فهي بذلك روح علم الحاسب. و سميت الخوارزمية بهذا الاسم نسبة إلى العالم ابو جعفر محمد بن موسى الخوارزمي.
- **البرنامج:**
- هي توصيف لخوارزمية حل مسألة معينة بإحدى لغات البرمجة التي يقبلها الحاسوب.
- **لغة البرمجة**:
- هي مجموعة من المفردات والقواعد والدلالات المعرفة التي تسمح بكتابة برنامج يمكن تنفيذه على الحاسوب.
- **المترجم (Compiler):**
- هو برنامج يفهم البرنامج المكتوب بلغة برمجة معينة ويحوله إلى برنامج مكافئ مكتوب بلغة المجمع (Assembly) الخاصة بالمعالج الصغري للحاسوب (Microprocessor).
-

أهمية الخوارزمية

١. أثناء تصفحك للانترنت هناك الكثير من البيانات التي يتم نقلها بينك وبين مزود هذه البيانات، وتستخدم الخوارزميات في تحديد الطريق الأقصر التي يجب على هذه البيانات أن تسلكه.

٢. أثناء بحثك في محرك بحث معين، تستخدم الخوارزميات في إعطاءك ناتج البحث الأفضل وبسرعة وكفاءة عالية.

٣. تستخدم الخوارزميات أيضاً في الألعاب (مثل الشطرنج)، حيث تستطيع الخوارزمية تحديد الحركة الأفضل لتستطيع التغلب عليك!

٤. تخيل أنك في رحلة سياحية وتريد أن تزور ستة أماكن، يمكن استخدام الخوارزميات في تحديد الطريق الأقصر لزيارة الأماكن الستة التي تريد زيارتها .

- إذاً، هناك فوائد كبيرة للخوارزميات ومجالات استخدام واسعة لها

لكل مشكلة عدة خوارزميات

- عندما تواجهنا مشكلة ما ونريد حلها ، سيكون لدينا العديد من الطرق (الخوارزميات) للحل ، ولكننا طبعاً نحتاج إلى الحل الأمثل الذي يحتاج إلى خوارزمية مثالية ، ولكي تكون الخوارزمية مثالية يجب أن تكون واضحة وبسيطة وذات فعالية عالية.
- لنأخذ مثالاً بسيطاً :
- هناك عدد غير منته من الطرق للوصل بين نقطتين على مستوٍ ، ولكن أقصر هذه الطرق هو الطريق المستقيم بينهما.

مواصفات الخوارزمية الجيدة

- يجب أن تكون منتهية وتنتهي بعدد منتهي من التعليمات أو العمليات.
- يجب أن تكون محددة ودقيقة بمعنى أن كل تعليمة يجب أن توصف بدون التباس.
- يجب تحديد مجال تعريف معطيات الدخل إن وجدت
(متحولات أعداد صحيحة ،حقيقية ،أحرف ،سلسلة نصية .....).
- يجب أن تكون هنالك نتيجة واحدة على الأقل.
- يجب أن تكون فعالة ،أي أن تكون العمليات كلها قابلة للتنفيذ وفي وقت منته من قبل شخص يستخدم الإمكانات اليدوية.

عامل تقييم الخوارزميات

- للمقارنة بين عدة خوارزميات خاصة لمشكلة معينة وتحديد الأمثل فيما بينها نقوم بحساب تعقيد الخوارزمية وهو عبارة عن طريقة منهجية لقياس كلفة هذه الخوارزمية من حيث الذاكرة والسرعة والدقة في النتائج.

- مثال :
- للمقارنة بين محركي البحث google, yahoo يمكننا الاعتماد على دقة نتائج البحث ومطابقتها للاستعلام إضافة لسرعة الإجابة .

تطوير الخوارزمية

١. التعرف على المشكلة بشكل واضح .
٢. تحديد الدخل(المعلومات التي تحتاجها لحل المشكلة) و الخرج (النتيجة التي تريد أن تحصل عليها من الخوارزمية).
٣. تحدد الخطوات التي ستعالج الدخل للحصول على الخرج .
٤. اختبار الخوارزمية

طرق التعبير عن الخوارزميات

١. الطريقة النصية (شبه ترميز)
٢. الطريقة الصورية (المخططات التدفقية)

الطريقة النصية (شبه الترميز)

- يمكن التعبير عن الخوارزمية باستخدام احدى اللغات الطبيعية (عربي ،فرنسي..) ولكن هذه اللغات تتميز بالغموض لذلك نحن بحاجة لاستخدام طريقة منتظمة للتعبير عن الخوارزمية بشكل يتيح لنا حرية التعبير عن الحل مع الاحتفاظ بسهولة نقل الحل إلى لغة برمجة يفهمها الحاسوب ،سنسمي تجاوزا هذا الطريقة في التعبير عن الخوارزمية لغة الخوارزمية.

- التعليمات أو الأوامر الأساسية الخمسة التالية:

تعليمة القراءة ،تعليمة الكتابة ،تعليمة الإسناد ،التعليمة الشرطية ،التعليمة التكرارية.

١. تعليمة القراءة

- قراءة قيمة من الدخل (لوحة المفاتيح) لوضعها في الذاكرة ،شكلها:

❖ **اقرأ < اسم المتحول >**

- مثلا:
- اقرأ a // ضع القيمة التي تعطى من الدخل في المتحول a .
- اقرأ a,b // ضع القيمتين المعطاتين من الدخل في المتحولين a,b على الترتيب.

٢. تعليمة الكتابة:

- وهي كتابة قيمة معينة على وحدة الإخراج (الشاشة) ،شكلها:

اكتب < صيغة >

- مثلا:
- اكتب a // اكتب أو ضع محتوى المتحول a على وحدة الخرج.
- لتكن قيمة a=130
- اكتب a+3 // اكتب قيمة الصيغة الحسابية (يظهر على الشاشة: العدد ١٣٣)

٣. تعليمة الإسناد:

- وهي إسناد قيمة صيغة لمتحول (خانة في الذاكرة) ،شكلها:

< صيغة > ← < اسم المتحول >

- مثلا:
- ليكن لدينا المتحولين a,b والقيم المخزنة فيهما (a=10,b=50) ولننفذ التعليمات التالية:
- $\begin{cases} A \leftarrow 70 \\ B \leftarrow A \end{cases}$ // تصبح قيمة الخانتين بعد الإسناد (a=70,b=70) .

٤. التعليمة الشرطية:

- وفيها يجري التنفيذ بعد اختبار شرط ،وترد بأحد الشكلين:
- **إذا < شرط > نفذ**
  | < مجموعة تعليمات >
- **إذا < شرط > نفذ**
  | < مجموعة تعليمات ١ >
  وإلا
  | < مجموعة تعليمات ٢ >

٥. تعليمة التكرار:

- تستخدم لتكرار مجموعة من التعليمات مادام شرط محدد محققا ،شكلها:
- **مادام < شرط > كرر**
- **| < مجموعة تعليمات >**
- علينا ملاحظة أن < مجموعة التعليمات > يجب أن تتضمن بالضرورة عمليات تساهم في تغيير متحولات الشرط < شرط > وإلا سندخل في حلقة لا نهائية.

تمارين

- ***التمرين الأول :***

نريد طباعة العدد الأكبر بين عددين يدخلهما المستخدم :

اقرأ a,b

إذا a>b نفذ

| اكتب a

وإلا

| اكتب b

- التمرين الثاني:
- أوجد القاسم المشترك الأعظم لعددين صحيحين موجبين.
- الحل :

اقرأ A, B

مادام (A لا تساوي B) كرر

| إذا $(A > B)$ نفذ

| $A \leftarrow A - B$

| وإلا

| $B \leftarrow B - A$

$GCD \leftarrow A$

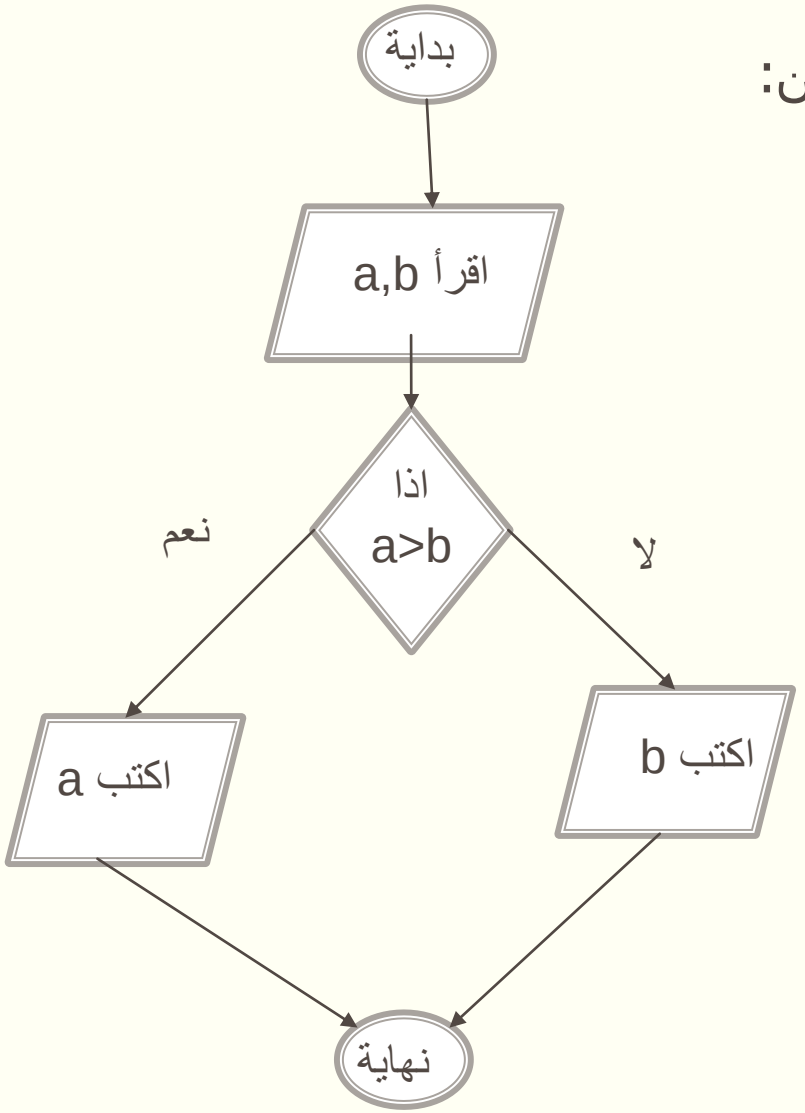
اكتب "القاسم المشترك الأعظم هو"، GCD

الطريقة البيانية

- تعتمد الطريقة البيانية لصياغة الخوارزميات على توضيح خطوات تنفيذ الخوارزمية باستخدام أشكال هندسية خاصة و أسهم تصل بينها إضافة إلى عبارات باللغة الطبيعية و تعابير رياضية أو منطقية. وبذلك نحصل على ما يسمى بالمخطط التدفقي Flowchart Diagramونلاحظ هنا أن الأسهم تفصل العمليات اللازمة لإنجاز العمل و تبين تسلسلها يستخدم المخطط التدفقي أشكالاً هندسية متفق عليها خصص كل منها لنوع من العمليات .

الشكل	الوظيفة
الدائرة أو الشكل البيضوي	لتحديد بداية الخوارزمية ونهايتها
المستطيل	للعمليات التنفيذية العادية أو الحسابية أو المنطقية
المعين	العمليات التي ترتبط باختبار تحقق شرط وتتطلب قرارا منطقيا
متوازي الأضلاع	عمليات الإدخال والإخراج
السهم	اتجاه التنفيذ

- تمرين : ارسم المخطط التدفقي لطباعة العدد الأكبر بين عددين:



وظائف

حل الخوارزميات التالية بالطريقتين السابقتين:

١. اكتب خوارزمية تحسب المتوسط الحسابي ل N عدد يدخله المستخدم.

٢. اكتب خوارزمية تحسب عاملي عدد معين يدخله المستخدم.

٣. اكتب خوارزمية تحسب عدد الأعداد الموجبة والسالبة في n عدد مدخل من المستخدم.

خوارزمية الفرز الفقاعي

المحاضرة الثانية
م. لمى السبع

مقدمة

- من أبرز و أهم العمليات التي يتم تنفيذها على البيانات المخزنة داخل تراكيب البيانات ما يعرف **ترتيب العناصر** و **البحث عن العناصر**.
- و ذلك لما لها من أثر إيجابي بالغ في العمل الاقتصادي.
- و قد تم تطوير العديد من تراكيب البيانات لجعل تنفيذ هذه العمليات أكثر سهولة و سرعة.

ما الهدف من خوارزميات الترتيب؟

- تهدف خوارزميات الترتيب إلى العمل على ترتيب مجموعة من العناصر المخزنة ضمن أحد أشكال تراكيب البيانات، و الترتيب يتم بناءً على كيفية محددة فمثلا تصاعدي، تنازلي، أبجدي، ...
- و تتميز خوارزمية عن أخرى من خوارزميات الترتيب في مدى تعقيد الخوارزمية من حيث الوقت و المساحة التخزينية المطلوبة.

خوارزمية Bubble Sort

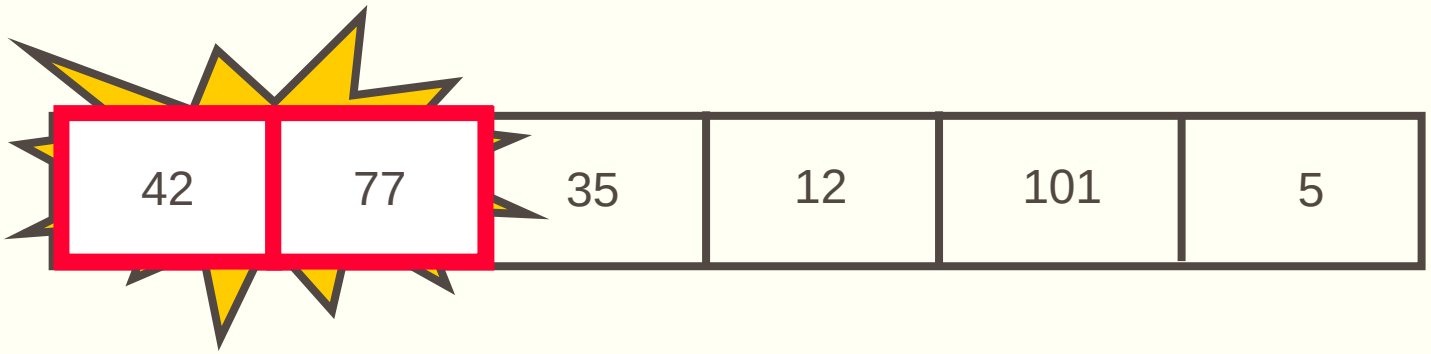
- تعتمد هذه الخوارزمية على مقارنة العناصر المتجاورة، وبناءً على هذه المقارنة يتم عمل الترتيب سواء تصاعدي أو تنازلي.
- و هذا النوع من الخوارزميات نحتاج فيه إلى المرور على البيانات المخزنة بعدد مرات يساوي عدد العناصر المخزنة مطروحا منه واحد (n-1)، و ذلك في أسوء الأحوال (ما المقصود بـ أسوء الأحوال؟)
- أسوأ الأحوال : في حال أردنا أن نرتب العناصر تصاعديا عندها تكون أسوأ الأحوال أن تكون العناصر بالترتيب التنازلي
- وأفضل الأحوال أن تكون هذه العناصر مرتبة تصاعديا مسبقا. لذلك لا نكون بحاجة لترتيبها.
- حيث n عدد العناصر المراد ترتيبها.

آلية عمل خوارزمية الفرز الفقاعي

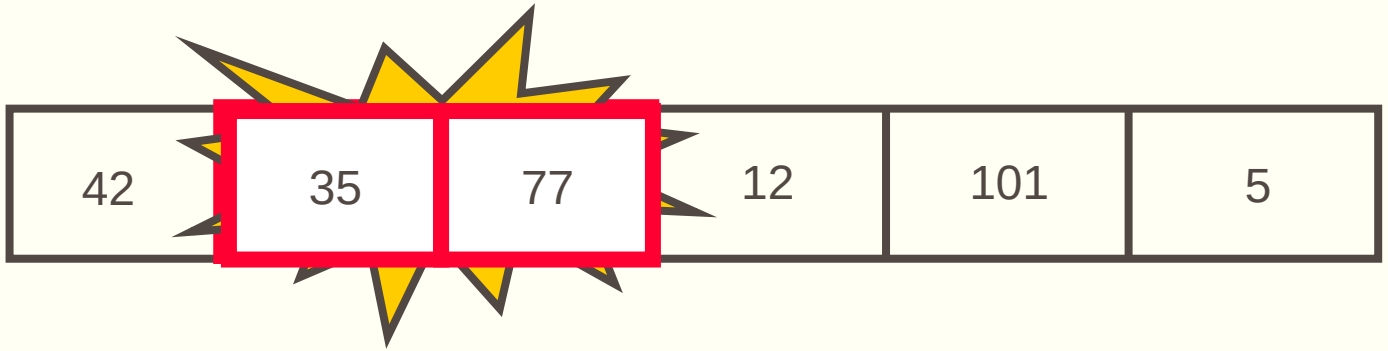
- لترتيب مجموعة من العناصر
  - ✓نبدأ من بداية المصفوفة لنهايتها
  - ✓في كل مرور نختار أكبر قيمة بين العددين ونضعها في فقاعة وننتقل بها على بقية العناصر .
- ويتم تكرار العملية السابقة حتى تصبح آخر قيمة في مكانها الصحيح

77	42	35	12	101	5

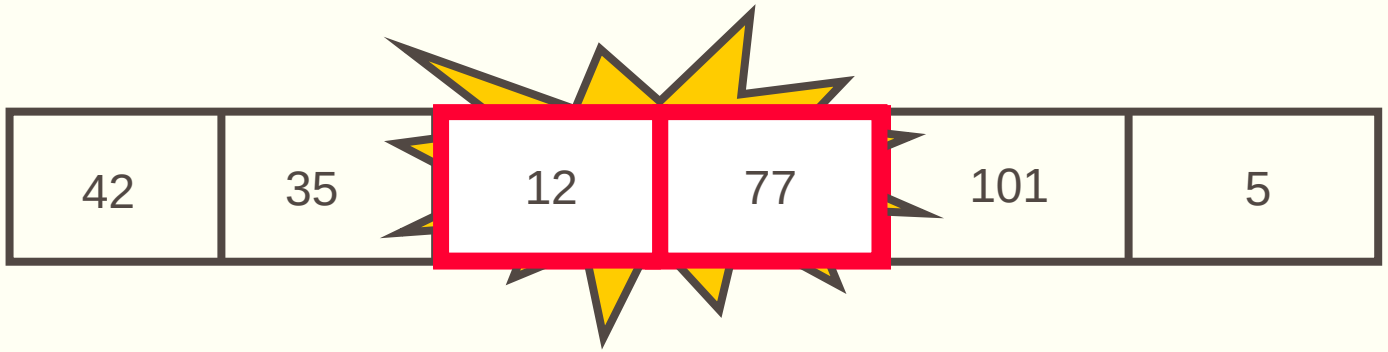
تتمة توضيح المثال



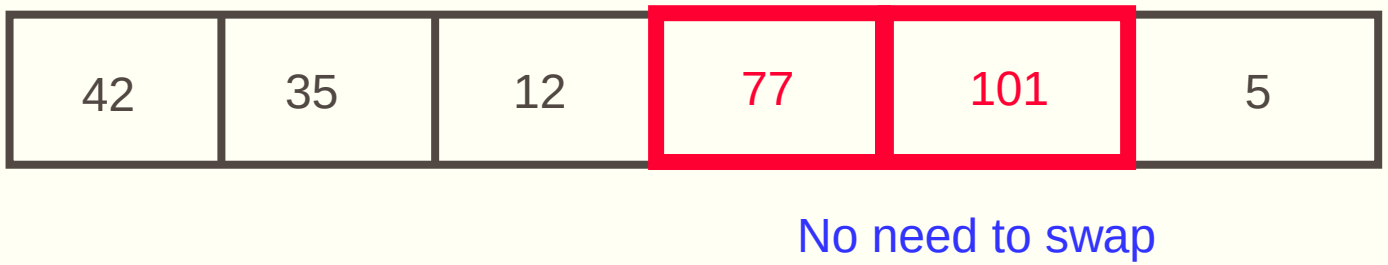
تتمة توضيح المثال



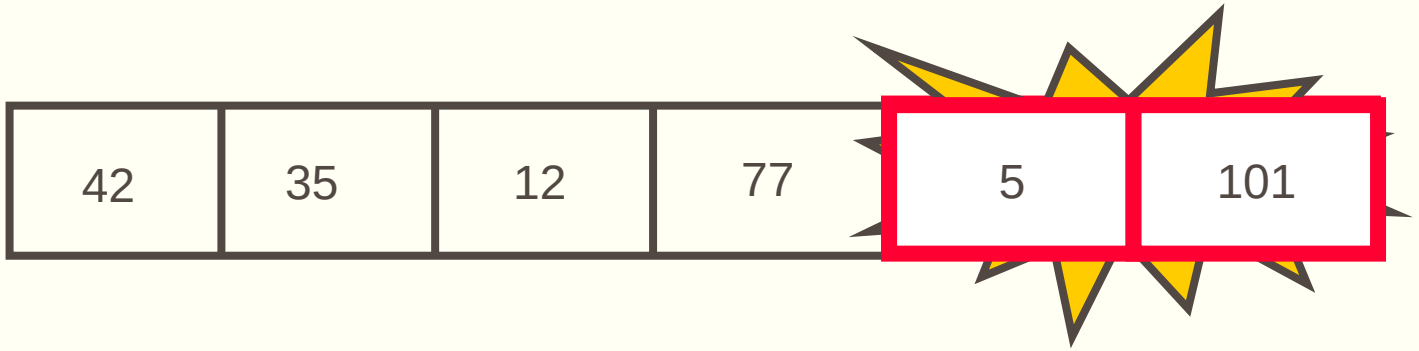
تتمة توضيح المثال



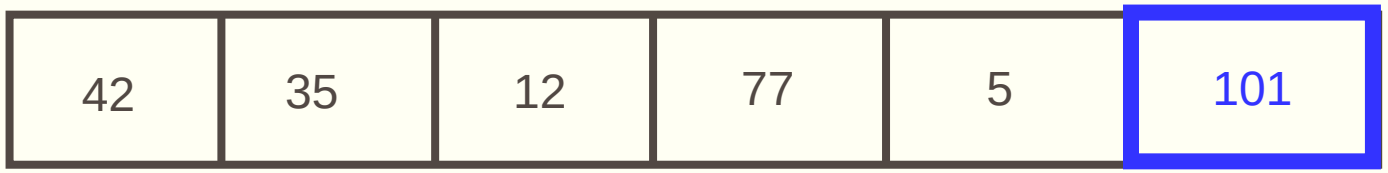
تتمة توضيح المثال



تتمة توضيح المثال

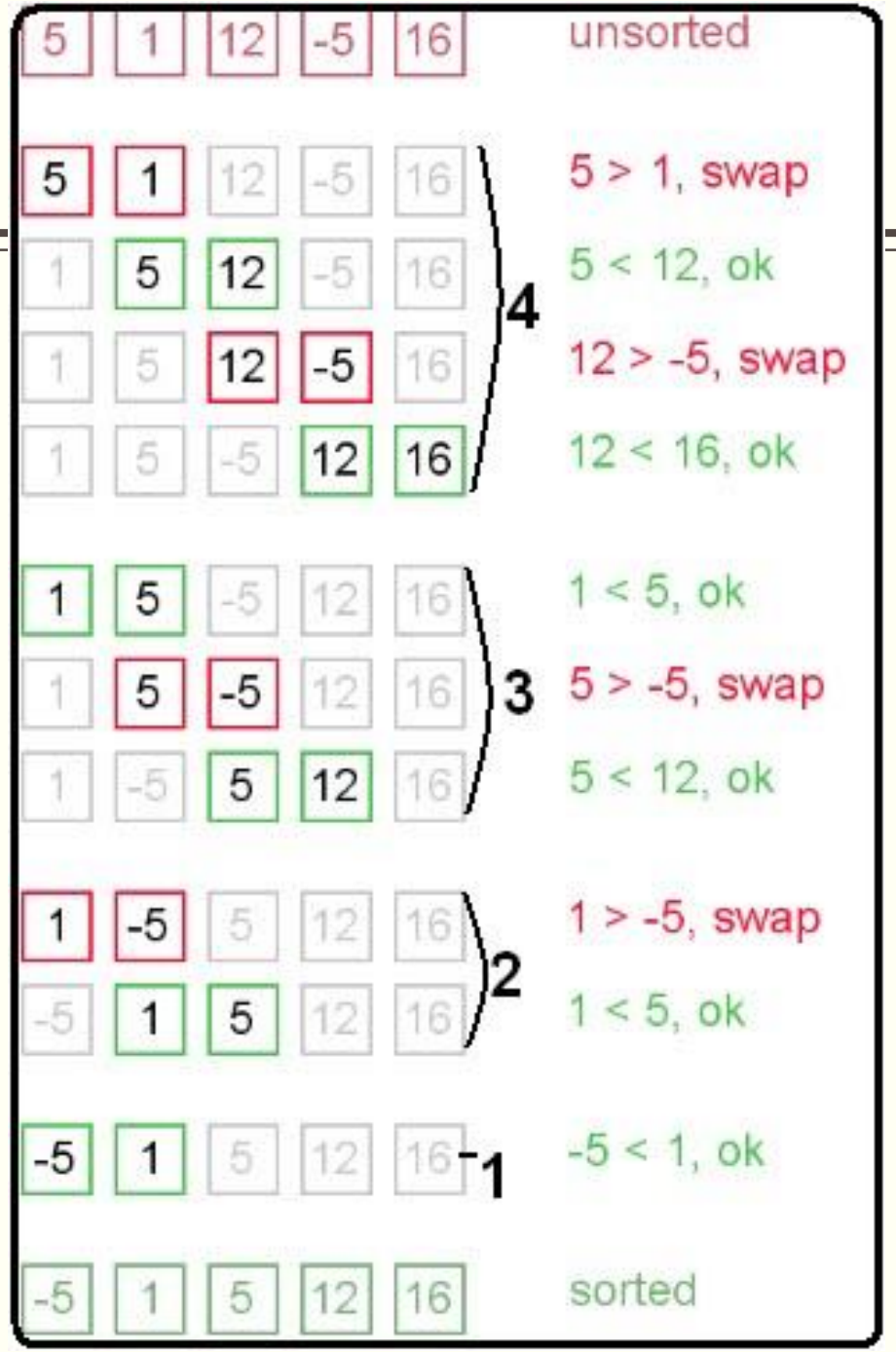


تتمة توضيح المثال



Largest value correctly placed

مثال توضيحي آخر



خوارزمية الفرز الفقاعي ..... شبه ترميز

- المدخلات : مصفوفة من العناصر حجمها n المطلوب ترتيب هذه العناصر تصاعدياً.
- نبدأ بداية المصفوفة ونقارن أول عنصر مع العنصر التالي

  إذا (عنصر ذو الدليل الأول > من العنصر التالي)

  أجر عملية التبديل بين العنصرين
- انتقل للعنصرين التاليين واعد نفس الخطوة السابقة n-1 مرة.

الكود البرمجي

```
1.    static void Main(string[] args)
2.          {
3.              int[] id = { 12, 30, 1, 4, 7, 2 };
4.              Console.Write(" Elements of Array Before Sorting ");
5.              for (int x = 0; x < id.Length; x++)
6.                  Console.Write("  " + id[x]);
7.              Console.WriteLine("  ");
8.              for (int i =0; i< id.Length; i++)
9.                  for (int j =0; j< id.Length-1; j++)
10.                     if (id[j] > id[j + 1])
11.                     {
12.                         int hold = id[j];
13.                         id[j] = id[j + 1];
14.                         id[j + 1] = hold; }
15.             Console.Write(" Elements of Array After Sorting ");
16.              for (int x = 0; x < id.Length; x++)
17.                 Console.Write("  "+id[x]); }
```

خوارزمية الفرز بالدمج

المحاضرة الثالثة
م. لمى السبع

خوارزمية merge Sort

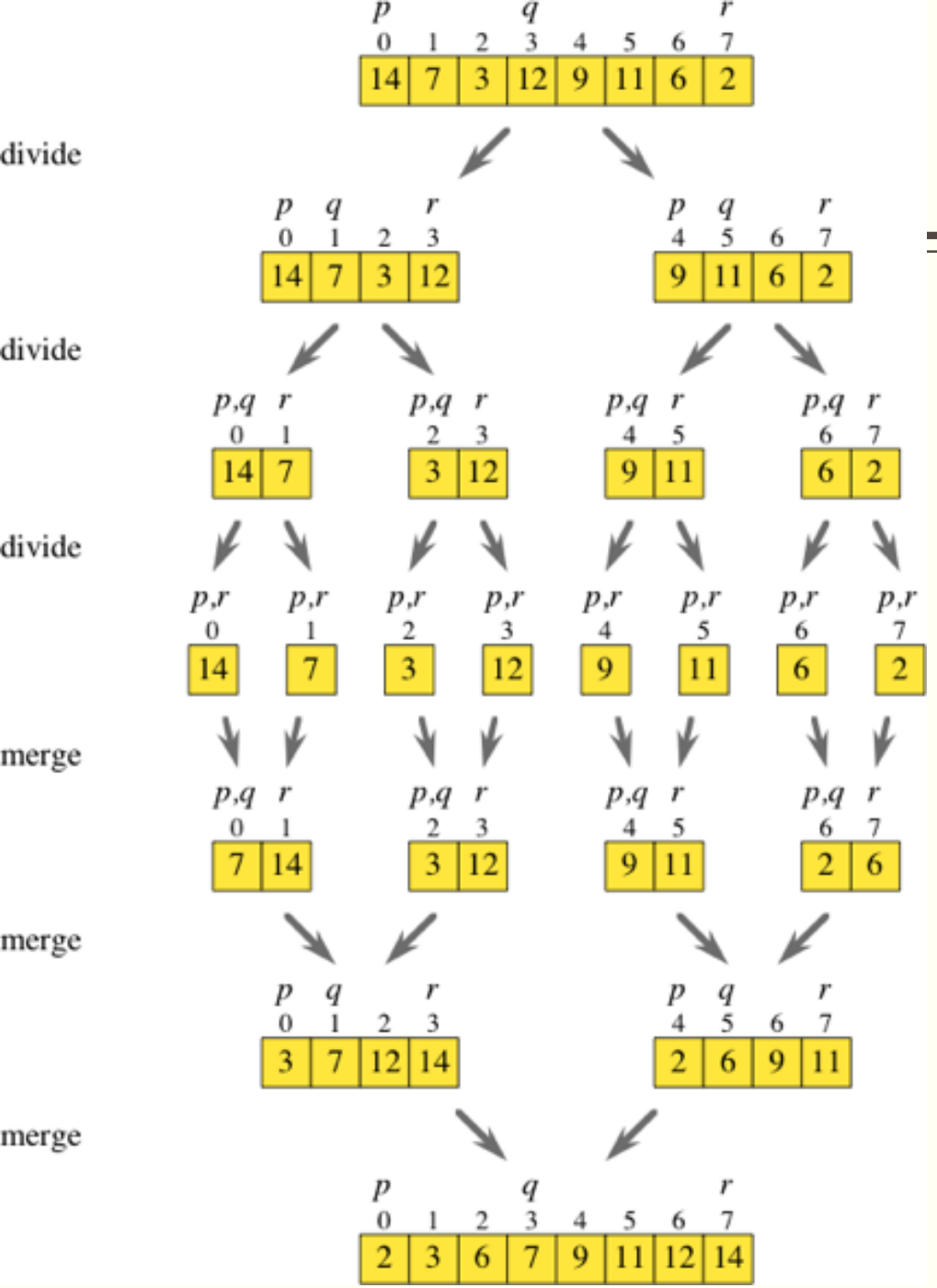
- **الهدف:** ترتيب قائمة a[1..n] ترتيبا تصاعدياً.
- آلية عمل الخوارزمية:

١. تقسم القائمة إلى قائمتين a[1,m] & a[m+1,n] وكل قائمة جزئية ترتب داخلياً.
٢. وأخيرا تدمج القائمتان الجزئيتان المرتبتان للحصول على قائمة مرتبة واحدة مرتبة.

❖ملاحظة :

❖تعتبر خوارزمية الفرز بالدمج أحد أنواع خوارزميات فرق-تجميع وهذا ما يوضحه المثال التالي.

مثال توضيحي



- p: هي بداية المصفوفة ونبدأ بالدليل رقم ٠
- R: دليل نهاية المصفوفة وفي المثال قيمتها ٧
- Q=(p+r)/2 هي دليل منتصف المصفوفة حيث

يتم تقسيم المصفوفة الى مصفوفتين جزئيتين .

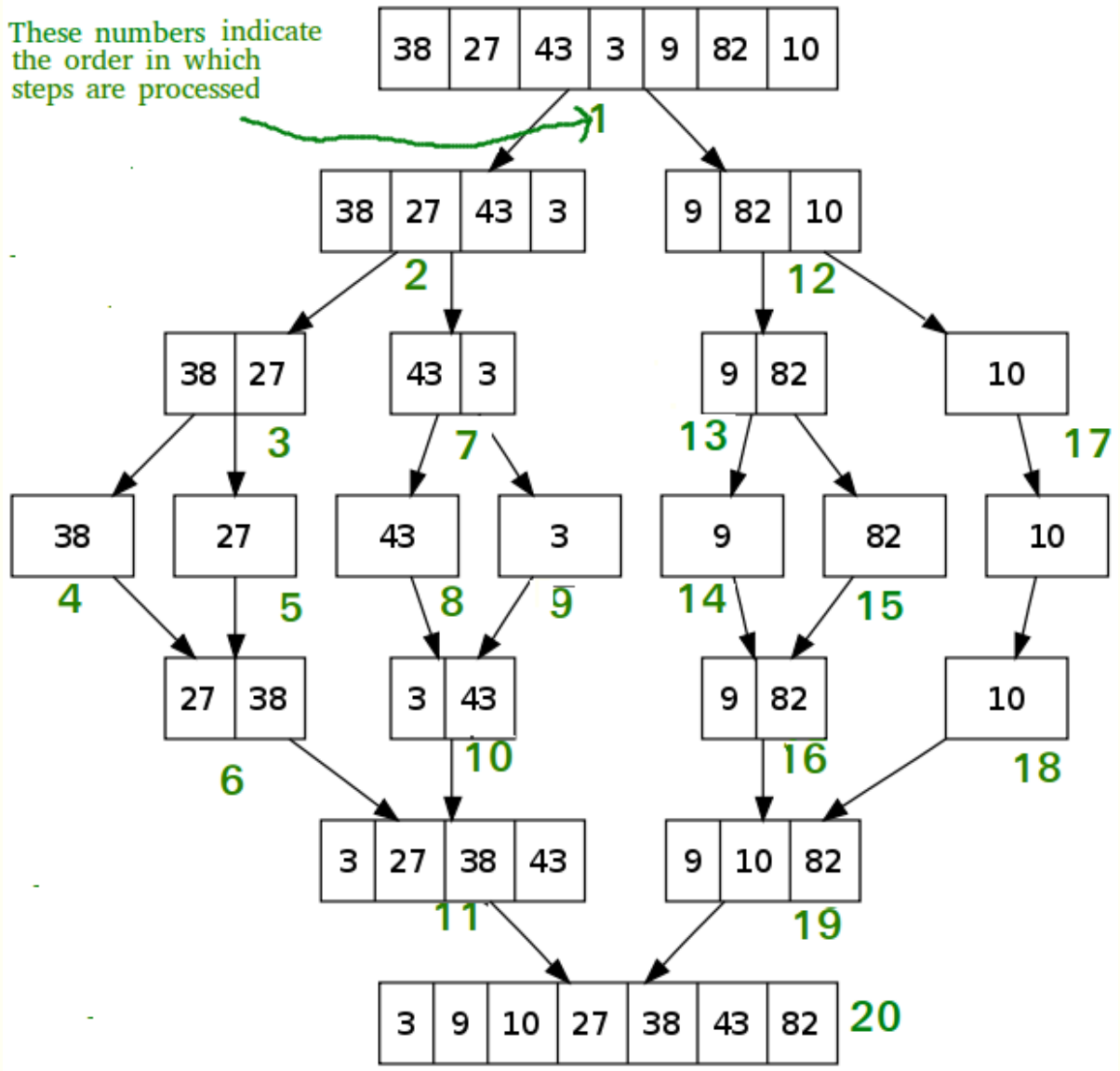
ونستمر بعملية التقسيم حتى تصبح قيمة p=r أي المصفوفة تتكون من عنصر واحد فقط. وهذه هي المرحلة الأولى من مراحل الخوارزمية وتدعى مرحلة فرق (تقسيم).

❖وفي مرحلة التجميع يتم تشكيل مصفوفات مرتبة اعتماداً على القيم الأحادية التي تم الوصول لها.

شبه الترميز

```
MergeSort(arr[], l,  r)
If r > l
     1. Find the middle point to divide the array into two halves:
             middle m = (l+r)/2
     2. Call mergeSort for first half:
             Call mergeSort(arr, l, m)
     3. Call mergeSort for second half:
             Call mergeSort(arr, m+1, r)
     4. Merge the two halves sorted in step 2 and 3:
             Call merge(arr, l, m, r)
```

مثال آخر مع توضيح تسلسل العمليات



الكود البرمجي لتابع mergeSort العودي

```
/* l is for left index and r is right index of the
   sub-array of arr to be sorted */
void mergeSort(int arr[], int l, int r)
{
    if (l < r)
    {
        // Same as (l+r)/2, but avoids overflow for
        // large l and h
        int m = l+(r-l)/2;

        // Sort first and second halves
        mergeSort(arr, l, m);
        mergeSort(arr, m+1, r);

        merge(arr, l, m, r);
    }
}
```

الكود البرمجي لتابع merge

```
// Merges two subarrays of arr[].
// First subarray is arr[l..m]
// Second subarray is arr[m+1..r]
void merge(int arr[], int l, int m, int r)
{
    int i, j, k;
    int n1 = m - l + 1;
    int n2 =  r - m;

    /* create temp arrays */
    int L[n1], R[n2];

    /* Copy data to temp arrays L[] and R[] */
    for (i = 0; i < n1; i++)
        L[i] = arr[l + i];
    for (j = 0; j < n2; j++)
        R[j] = arr[m + 1+ j];

    /* Merge the temp arrays back into arr[l..r]*/
    i = 0; // Initial index of first subarray
    j = 0; // Initial index of second subarray
    k = l; // Initial index of merged subarray
    while (i < n1 && j < n2)
    {
        if (L[i] <= R[j])
        {
            arr[k] = L[i];
            i++;
        }
        else
        {
            arr[k] = R[j];
            j++;
        }
        k++;
    }
```

```
    /* Copy the remaining elements of L[], if there
       are any */
    while (i < n1)
    {
        arr[k] = L[i];
        i++;
        k++;
    }

    /* Copy the remaining elements of R[], if there
       are any */
    while (j < n2)
    {
        arr[k] = R[j];
        j++;
        k++;
    }
```

وظيفة

- المطلوب

١. اجراء بحث عن خوارزمية الفرز بالحشر (الاقحام)
٢. اجراء بحث عن خوارزمية الفرز بالاختيار

خوارزميات البحث

المحاضرة الرابعة
م. لمى السبع

خوارزمية البحث الخطي

- خوارزمية البحث الخطي هي إحدى خوارزميات البحث التقليدية و الأساسية، إذ تُعتبر طريقة للبحث عن **موقع** قيمة معينة داخل مصفوفة، و أساسيتها تنبع من إتباعها منهجية بسيطة جداً، فهي تُنجز عملية البحث بالتحقق من كل عنصر بصورة متسلسلة أو كما يروق لمتخصصي الخوارزميات (خطية). تبدأ من بداية المصفوفة و حتى نهايتها.

طريقة عمل خوارزمية البحث الخطي

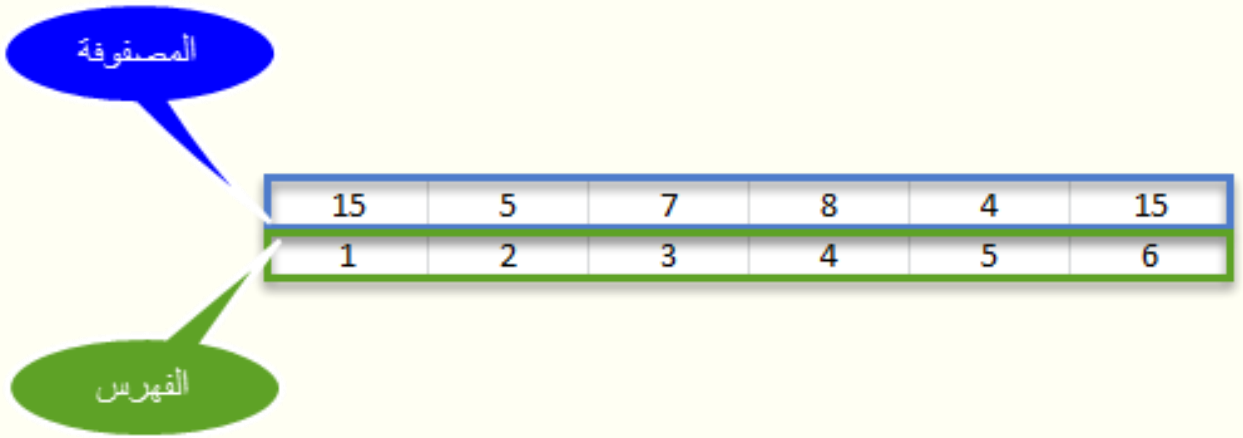
- لإجراء البحث الخطي لا بد من توفر عناصر أساسية و هي مصفوفة البحث و العنصر المُراد البحث عنه.
- بعكس كثير من خوارزميات البحث، لا يتطلب أن تكون مصفوفة البحث مرتبة ترتيباً تصاعدياً أو تنازلياً، هذا لا يعني أن المصفوفة إذا كانت مرتبة فإنها تُخل بأساسيات البحث الخطي، على العكس تماماً، فكل أنواع المصفوفات مرتبة أو غير مرتبة مُرحبٌ بها في خوارزمية البحث الخطي.
- مصفوفة البحث لها فهرس (Index)، يبدأ من الواحد أو الصفر، لا تختلف كثيراً فالنتيجة واحدة.
- يبدأ البحث بالتحقق من العنصر الأول هل هو مساوٍ للعُنصر المراد البحث عنه، إذا وُجد العنصر المراد البحث عنه يُحتفظ برقم الفهرس الخاص بالعنصر و تُنهى عملية البحث.
- إذا لم يُوجد العنصر المراد البحث عنه تنتقل عملية البحث من العنصر الأول في المصفوفة إلى العنصر الثاني ثم الثالث و هكذا إلى أن يكتمل البحث عن العُنصر المراد البحث عنه في كامل المصفوفة.
- يُرمز لحالة عدم إيجاد العُنصر المراد البحث عنه بقيمة مختلفة عن قيم فهرس المصفوفة، مثلاً إذا كانت المصفوفة بطول ١٠ عناصر، و تبدأ من ١ إلى ١٠، فيُرمز إلى حالة عدم إيجاد العنصر المراد البحث عنه بالرقم (٠)، لاحظ أنه كما ذُكر سابقاً في النقطة رقم ٣، في بعض المصفوفات قد يبدأ الفهرس من الرقم **صفر**، في هذه الحالة قد يُلجأ إلى الإشارة بالقيمة NULL في حالة عدم إيجاد العنصر المراد البحث عنه.

مثال لطريقة البحث في خوارزمية البحث الخطي

- نفترض أن لدينا المصفوفة أدناه حيث :

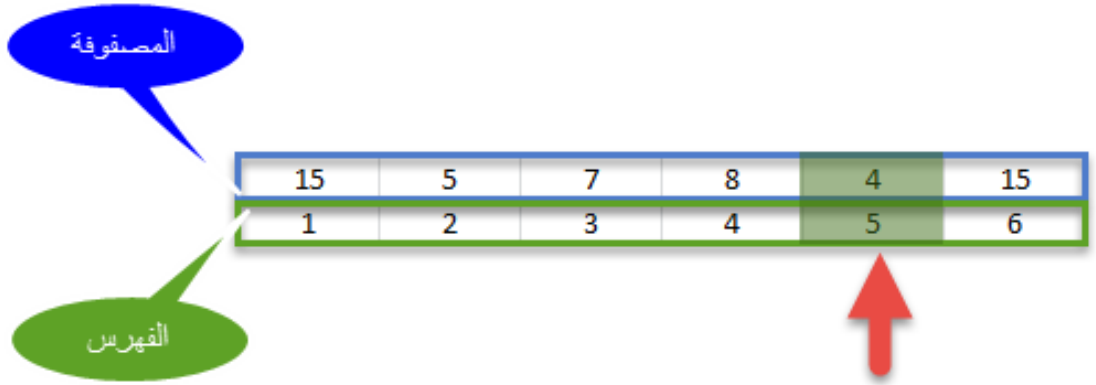
❖عدد عناصر المصفوفة n=6

❖القيمة المراد البحث عنها x=4



آلية الحل:

- تبدأ عملية البحث بمقارنة العنصر بالخانة رقم 1 حيث j=1، هل A[j]=x؟ لا. إذن ننتقل إلى الخطوة الثانية.
- مقارنة العنصر بالخانة رقم 2 حيث j=2، هل A[j]=x؟ لا. إذن ننتقل إلى الخطوة الثالثة.
- مقارنة العنصر بالخانة رقم 3 حيث j=3، هل A[j]=x؟ لا. إذن ننتقل إلى الخطوة الرابعة.
- مقارنة العنصر بالخانة رقم 4 حيث j=4، هل A[j]=x؟ لا. إذن ننتقل إلى الخطوة الخامسة.
- مقارنة العنصر بالخانة رقم 5 حيث j=5، هل A[j]=x؟ نعم. إذن هنا نحتفظ بالرقم 5 و تُعتبر هذه نتيجة البحث.



الكود البرمجي

خوارزمية البحث الثنائي Binary Search

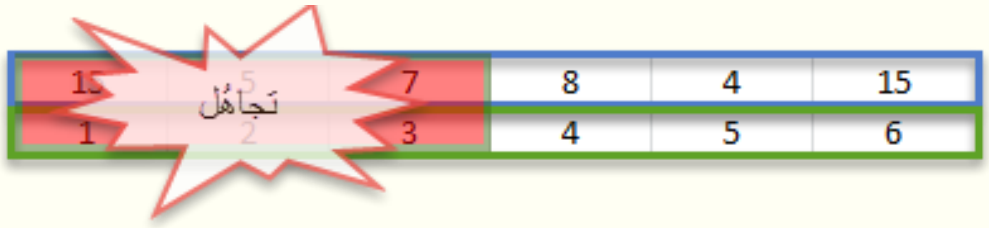
- كغيرها من الخوارزميات لا بد من توافر المدخلات المطلوبة و التي تتمثل في العُنصر المراد البحث عنه و المصفوفة التي سيتم البحث فيها
- يجب أن تكون المصفوفة مرتباً تصاعدياً (من الأصغر إلى الأكبر)، في حال لم تكن المصفوفة مرتبة تصاعدياً يجب عليك أن تُرتبها تصاعدياً حتى تستطيع تطبيق خوارزمية البحث الثنائي
- مصفوفة البحث لها فهرس ((Indexيبدأ من الصفر أو الواحد، لكليهما نفس النتيجة

آلية عمل الخوارزمية

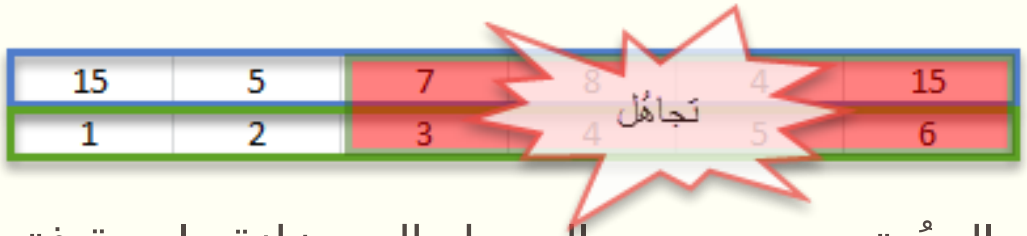
- بدأ الخوارزمية عملها بوضع مؤشر الفهرس في وسط المصفوفة حتى تقسم المصفوفة إلى قسمين، و هنا بداية الاستفادة الفعلية من مبدأ (فرّق تسُد)
- تتم مقارنة العنصر بالوسط (حيثُ المؤشر) بالعُنصر المُراد البحث عنه و النتيجة إحدى ثلاث حالات:

١. أن تساوي قيمة العُنصر بالمصفوفة قيمة العُنصر المُراد البحث عنه، و بالتالي يتم الإحتفاظ بقيمة مؤشر الفهرس و يُوقف البحث.

٢. ن تكون قيمة العُنصر المراد البحث عنه أكبر من قيمة العنصر حيث المؤشر، في هذه الحالة يتم تجاهل جميع عناصر النصف الأيسر من المصفوفة و يُصبح الجزء الأيمن من المصفوفة مُدخلاً جديداً لعملية بحث عن طريق خوارزمية البحث الثنائي.

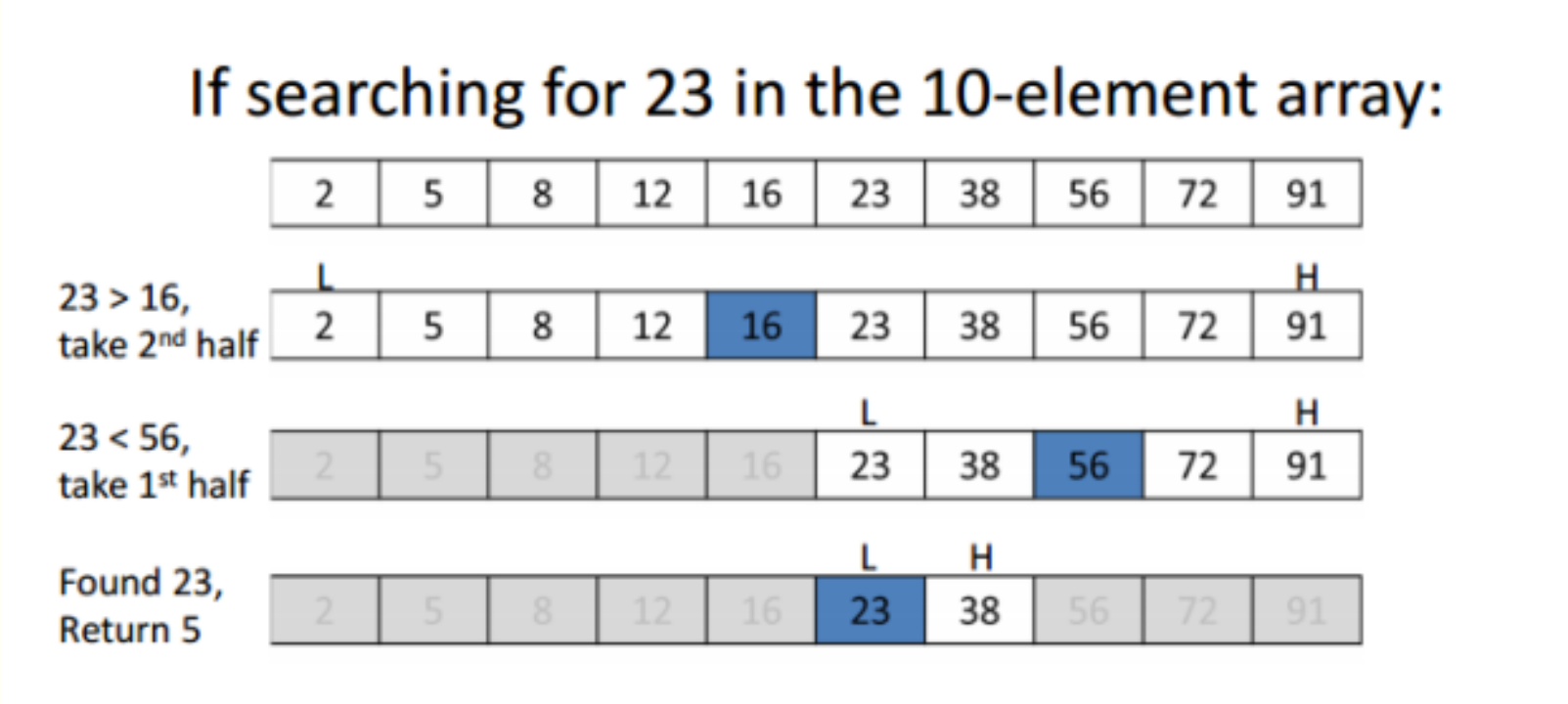


٣. أن تكون قيمة العُنصر المراد البحث عنه أقل من قيمة العنصر حيثُ المؤشر، في هذه الحالة يتم تجاهل جميع عناصر النصف الأيمن من المصفوفة و يُصبح الجزء الأيسر من المصفوفة مُدخلاً جديداً لعملية بحث عن طريق خوارزمية البحث الثنائي



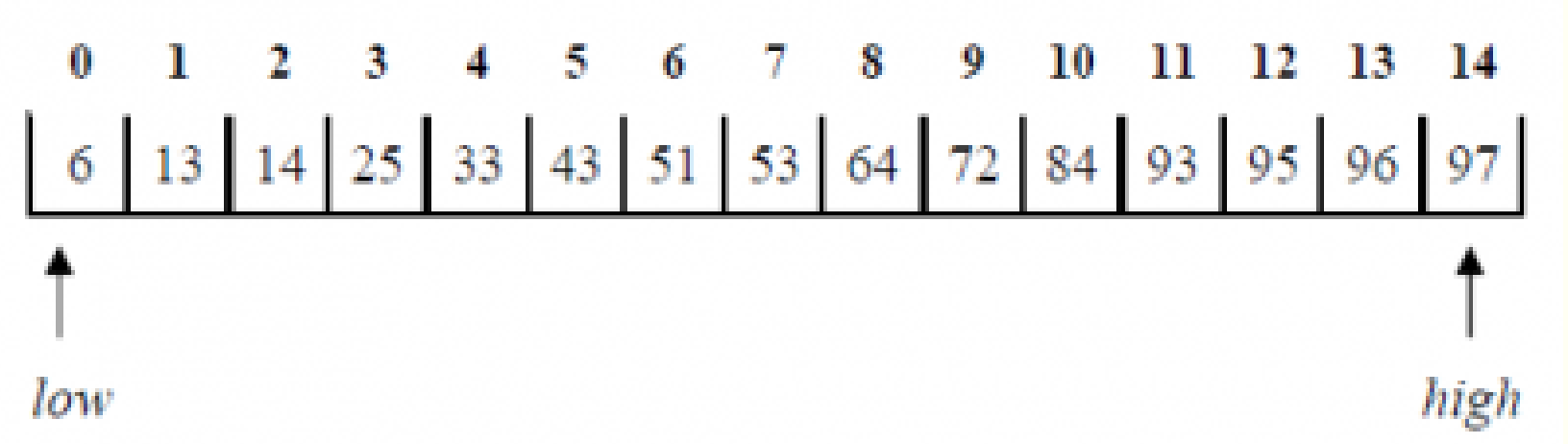
- تُكرر عملية البحث في النصف المتُبقي حتى يتم الوصول إلى خانة واحدة فقط تُمثل موقع العنصر المُراد البحث عنه

مثال


If searching for 23 in the 10-element array:
2 5 8 12 16 23 38 56 72 91
23 > 16, take 2nd half
L H
2 5 8 12 16 23 38 56 72 91
23 < 56, take 1st half
L H
2 5 8 12 16 23 38 56 72 91
Found 23, Return 5
L H
2 5 8 12 16 23 38 56 72 91

مثال آخر

المطلوب : البحث عن القيمة ٣٣

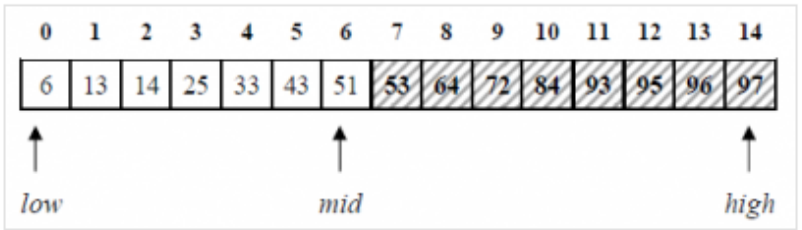


1. نحدد منتصف المصفوفة:

$$Mid = \frac{low + high}{2} = \frac{(0 + 14)}{2} = 7$$

2. تقسيم المصفوفة إلى قسمين وفقا لموقع mid:

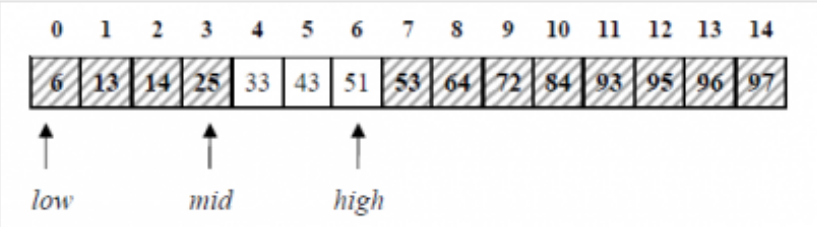
3. بما أن 33 أصغر من 53, فعملية البحث ستنفذ على النصف الأول:



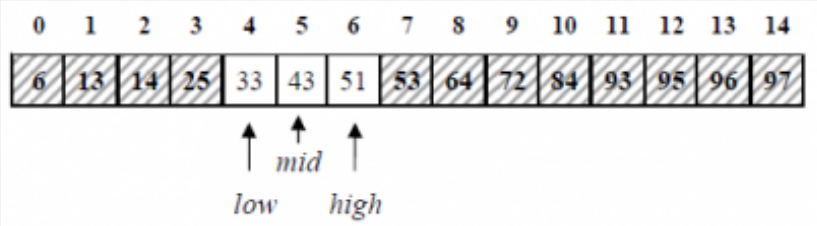
4. نعيد عملة التقسيم على النصف الأول و تصبح قيمة mid تساوي:

$$Mid = \frac{(0 + 7)}{2} = 3$$

5. بما أن 33 أكبر من 25, سيتم استبعاد النصف الأول و تستمر عملية البحث في النصف الثاني:



6. من جديد, نعيد عملية التقسيم حيث تصبح قيمة mid تساوي 5:



7. و أخيرا نجد أن القيمة 33 موجوده في الموقع low و الذي يساوي 4.

الكود البرمجي

```cpp
#include <iostream>
using namespace std;

int binarySearch(const int array[], int key, int low, int high) {
    int mid;
    if (high < low) return -1;
    else {
        mid = (low + high) / 2;
        if (array[mid] > key)
            return binarySearch(array, key, low, mid - 1);
        else if (array[mid] < key)
            return binarySearch(array, key, mid + 1, high);
        else
            return mid;
    }
}

int main() {
    const int LENGTH = 1000, centre = LENGTH / 2;
    int result, X[LENGTH];
    for (int i = 0, j = -centre; j <= centre; i++, j++)
        X[i] = j;
    result = binarySearch(X, -500, 0, LENGTH - 1);
    if (result != -1)
        cout << "la clef se trouve dans la case numéro " << result << endl;
    else
        cout << "Valeur introuvable" << endl;
    return 0;
}
```