الجامعة المستنصرية /كلية التربية / قسم علوم الحاسبات

Data Structures / هياكل بيانات

المرحلة الثانية

مقدمة :

يمكن تعريف هياكل البيانات بانها : دراسة طرق تمثيل المبرمجين للبيانات وعلاقة البيانات وعلاقة البيانات بالاجهزة (وخصوصا ذاكرة الحاسوب التي تخزن فيها البيانات) فيها كل البيانات تشمل طرق تنظيم المعلومات , والخوارزميات الكفؤة في الوصول لها وطرق التعامل معها او تداولها (كالاضافة والحذف والتحد يث والترتيب والبحث ....والخ)

لذا فان الاهتمام لاينحصر فقط باساليب الخزن وخوارزمية وانما قياس كلفة كل اسلوب من تلك الاساليب ومدى ملائمة استخدامها في الحالات المختلفة .

انواع هياكل البيانات:

توفر لغات البرمجة الصيغ المناسبة لتعريف واستخدام العناصر البيانية ذات القيمة الواحدة (المنفردة) فمثلا في لغة C++ تستخدم التعريفات.

```
Int     x  ;
Float   y  ;
Char    A  ;
```

لتمثل في ذاكرة الحاسوب ويتم التعامل معها بصيغ برمجية بسيطة مثلا :

```
X = 100;
Y =2.3;
A = 'B';
```

اما بالنسبة للعناصر البيانية التي تتكون من عدة قيم بيانية فانها تحتاج لاستخدام هياكل بياني مختلف وفيما يلي ذكر لاهم تللك الهياكل البيانية :

1- المصفوفة Array

2- القيد والسجل Struck

3- الملف File

4- الهياكل الخطية Linear Structures

4.1 الهياكل غير الموصولة Non-Linked Structures

* المكدس Steak

* الطابور Queue

* الطابور الدائري Circular Queue

4.2 الهياكل الموصولة Linked structures

* المكدس الموصول Linked Stack

* الطابور الموصول Linked Queue

5- الهياكل غير الخطية Non – Linear Structures

* المخططات Graphs

* المخطط المتجه Directed Graph

* هيكل الشجرة Tree Structure

* المخطط غير المتجه Un Direct Graph

<u>كيفية أختيار الهيكل البياني المناسب :</u>

لكل مجموعة من البيانات هنالك اكثر من طريقة لتنظيمها ووضعها في هيكل بياني معين ويتحدد ذلك وفق عدد من العوامل والاعتبارات لاختيار الهيكل البياني المناسب وهي :

1. حجم البيانات
2. سرعة وطريقة استخدام البيانات
3. السعة الخزنية المطلوبة
4. الزمن اللازم لاسترجاع اية معلومة من الهيكل البياني
5. اسلوب البرمجة

المصفوفة Array

المصفوفة : .هي عبارة عن مجموعة من المواقع الخزنية في الذاكرة تستخدم وتتصف بما ياتي :

1. جميع المواقع تكون من نوع بياني واحد ،حسب صيغة التعريف .....float ,int ,char
2. يمكن الوصـــول عشـــوائيا (Randomly Accessed) الى اي موقع من مواقعها دون الاعتماد على اي موقع في المصـفوفة فمقدار الوقت المطلوب للوصـول الى اي موقع هو مقدار ثابت.
3. مواقع عناصر المصفوفة تبقى ثابتة ولاتتغير اثناءالتعامل مع اي من عناصر المصفوفة.
4. تمثل المصفوفة في مواقع متعاقبة في الذاكرة.

تمثيل المصفوفة الاحادية في الذاكرة :

في لغة C++ تعرف هذه المصفوفة كالاتي :

int {**or any other type**} x [N]

Ex:

int A[10];

float A[20];

وهذا يعني تعريف هيكل بياني يسـتوعب مجموعة من العناصـر البيانية عددها N مثلا باسـم بياني واحد هي X ويســـتخدم الدليل (Index) للوصـــول الى العناصـــر البياني المطلوب وتتراوح قيمة الدليل (1=<i<=n) وبموجب هذا التعريف يحدد مترجم اللغة (Compiler) المنطقة الجزئية لاستيعاب مجموعة العناصر البيانية ويكون المرقم الاول مخصصا للعنصر الاول في المصـــفوفة وهو ما يطلق عليه عنوان البداية (Base Address) (BA) وليكن افتراضـا هو (500) اما العنصـر الثاني للمصـفوفة فيكون عنوانه بعد عنوان البداية مباشـرة اي (501) وهكذا بقية العناصربالتتابع.

Location (x[i]) = Base Address + (i-1)

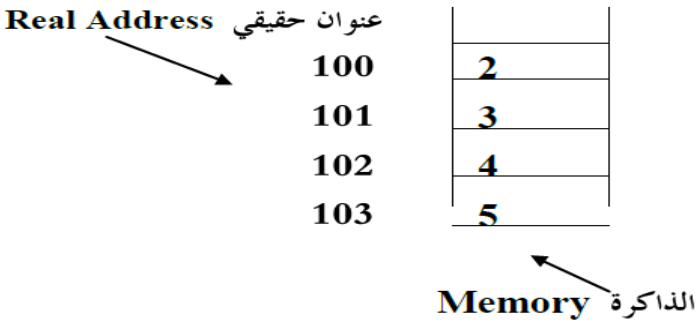
Ex:

A[4]

A=[2 3 4 5]

A[0]=2 , A[1]=3 , A[3]=4 , A[4]=5

عناوين نسبية وليست حقيقية



Ex: Let int x [n] any Array compute the Location of the element x [4] when the bass Address is 500?

Solution:

i=4

Location (x [4]) =500+ (4-1)

=500+3

=503

* فعندما يتضمن البرنامج اية اشارة او تعامل مع عناصر المصفوفة في اي ايعاز مثل cin>>x[i] و cout<< x[i] او غيرهما فان المترجم يعتمد العلاقة المشار اليها لتحديد الموقع المطلوب .

تمثيل المصفوفة الثنائية في الذاكرة :

في لغة C++ تعرف هذه المصفوفة كالاتي ،

int {or any other type} A[M][N]

Ex:

Int A[10][10];

Float A[20][20];

وهذا يعني تعريف هيكل بياني اسمه (A) يستوعب مجموعة من العناصر البيانية عددها (M*N) ويستخدم دليلين للوصول الى العناصر البياني المطلوب وهما

1=<i<=M لتحديد الصف الذي فيه العنصر

1=<j<=N لتحديد العمود الذي فيه العنصر

فمثلا العنصر A[3][5] حيث i=3 وj=5

وهذا يعني ان العنصر يقع في الصف الثالث والعمود الخامس .

طريقة خزن المصفوفة داخل الذاكرة:

Ex:

A[2][2]

A= $\begin{bmatrix} 2 & 3 \\ 5 & 6 \end{bmatrix}$

A[0][0]=2 , A[0][1]=3 , A[1][0]=5 , A[1][1]=6

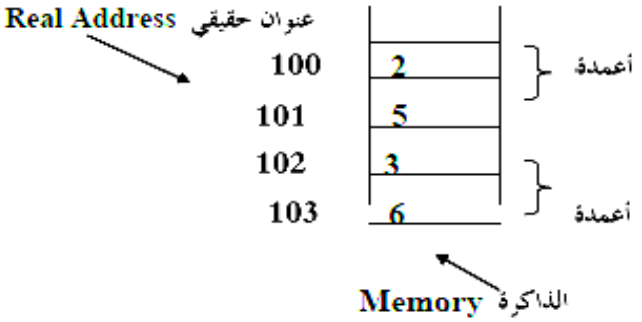
عناوين نسبية وليست حقيقية

التمثيل المنطقي للمصفوفة الثنائية: **Logical Structure**

$$A = \begin{pmatrix} a00 & a01 & \ldots\ldots & a0m \\ a10 & a11 & \ldots\ldots & a1m \\ \vdots & \vdots & & \vdots \\ an0 & an1 & \ldots\ldots & anm \end{pmatrix} \quad n*m$$

التمثيل الحقيقي للمصفوفة الثنائية: **Physical Structure**

(عمود – عمود):



ويعتمد مترجم اللغة (Compiler) احدى الطريقتين الاتيتين لتمثيل هذه المصفوفة :

أ- طريقة الاعمدة Column -wise method :

وهذه الطريقة مستخدمة في لغة الفورتران والبيسك.

اذ توخذ عناصر العمود الاول (j=1) للمصفوفة وتخزن في الذاكرة ابتداء من العمود الثاني ، وهكذا الى نهاية الاعمدة.

وعليه فان احتساب موقع العنصر A[i][j] يكون وفق العلامة التالية :

$$\text{Location (A [i] [j])} = \text{Base Address} + M*(j-1) + (i-1)$$

EX: Let int s [6] [8] any Array compote the location of the element s [5] [7] when Base Address is (300).

Sol:

i=5, j=7, M=6, N=8

Location (s [5] [7]) =300+ 6(7-1) + (5-1) = 300+ 36+4 =440

ب- طريقة الصفوف Row-wise method:

وهذه الطريقة مستخدمة في لغة باسكال ،كوبول ، C++

حيث توخذ جميع عناصر الصف الاول (i=1) للمصفوفة وتخزن في الذاكرة ابتداء من موقع البداية (Base Address).

ثم توخذ جميع عناصر الصف الثاني (i=2) للمصفوفة وتخزن في الذاكرة ابتداء من الصف الثاني . وهكذا الى نهاية الصفوف.

ولهذا فان احتساب موقع العنصر A[i][j] يكون وفق العلامة التالية :

Location (A[i] [j]) = Base Address + N* (i-1) + (j-1)

Ex: Let int A [5] [7] Compute the location of the element A [4] [6] When the base Address is 900.

Sol:

i=4, j=6 , M=5 , N =7

Location (A [4] [6]) = 900+ 7 * (4 -1) + (6-1)

= 900 + 21 + 5 = 926

5- الخيوط الرمزية Strings:

عبارة عن هيكل بياني يتكون من مجموعة من العناصر التي هي عبارة عن رموز تتمثل بالحروف الأبجدية والأرقام والرموز الخاصة التي تتواجد على لوحة المفاتيح ويعلن عنها بالطريقة التالية:

a. char name - of - string[size];

b. char *name;

Ex:-

1) char name[35];

2) char *name;

*ابرز الدوال التي تطبق على الخيوط الرمزية والتي تتواجد ضمن المكتبة <string.h> :

1- strcpy(st1,st2); : دالة لاستنساخ خيط رمزي معين من خيط آخر ، وممكن تحديد عدد الرموز المراد استقطاعها وبالشكل التالي: strncpy(st1,st2,n); تمثل n عدد الرموز المستقطعة.

2- strcmp(st1,st2); : دالة للمقارنة بين الخيوط الرمزية.

3- strlen(st); : دالة لحساب طول الخيط الرمزي .

4- strcat(st1,st2); : دالة لدمج خيطين رمزيين بحيث تكون النتيجة في الخيط الرمزي الاول.وممكن تحديد عدد الرموز التي تستقطع من الخيط الرمزي الثاني وتدمج الخيط الاول بواسطة: strncat(st1,st2,n); .

**ابرز الدوال التي تطبق على الخيوط الرمزية والتي تتواجد ضمن المكتبة <ctype.h> :

1- isalnum(ch); : دالة لمعرفة الرمز اذا كان عبارة عن حرف ابجدي أو رقم ، ترجع هذه الدالة قيمة صفرية اذا كان الرمز لا يساوي قيمة ابجدية ولا رقم.

2- isalpha(ch); : تختبر الرمز اذا كان حرف ابجدي او لا .

3- islower(ch); : تختبر الرمز فيما اذا كان صغير ترجع قيمة والا ترجع صفر.

4- isupper(ch); : تختبر الرمز اذاكان كبير فانها ترجع قيمة.

5- isdigit(ch); : تختبره اذا كان رقم ترجع قيمة.

6- struper(); : تحول الحرف من صغير الى كبير.

7- strlwr(); : تحول الحرف من كبير الى صغير.

8- strinv(); : تعكس الخيط الرمزي.

Ex: write program to read string and write the middle character.

```
#include<iostream.h>
#include<string.h>
void main()
{
int i,l;
char st[30];
cout<<"enter string: ";
cin>>st;
 l=strlen(st);
i=l/2;
cout<<st[i];
}
```

Struct

السجل / القيد

هو هيكل بياني يستخدم ال Struct عندما نريد ان نعرف مجموعة من العناصر البيانية المترابطة وليست من نوع واحد ، حيث ان المصفوفة لاتحقق ذلك لانها مجموعة من المواقع الجزئية من نوع واحد.

التعريف والاعلان عن Struct

هنالك طريقتين لتمثيل struct في C++ وهما :

1- Struct name_struct
{
Members *** فكل عنصر في struct يسمى عضوا او عنصرا**
} ;

Ex:
struct Item
{
int number;
Char name;
} ;

Main ( )
{
Item x;
}

2- typedef struct
{
members ;
name_struct ;
}
Ex :
typedef struct
{
int x ;
Char A ;
float y ;
account;
}
main ()
{
account s ,k ;
}

* لاستدعاء عضوا في الـ struct لغرض اعطاءه قيمة او طباعته واجراء العمليات له يذكر اسم الـ struct و(dot) ثم اسم member : وكما في العلاقة التالية :

Name _struct . Name _member

Ex: Write a program to read student name and three Degrees and find the average using struct.

Sol:

```
#include < iostrem.h>
Struct student {   char name[10];
                   int d1, d2, d3;
                   float Av;
                };
Main ( )
{
      Student   recst;
            Cout <<" enter the name ";
            Cin >> recst. Name;
            Cout<< "enter the first deg";
            Cin >> recst. d1;
            Cout<< "enter the second deg";
            Cin >> recst.d2;
            Cout<<" enter the third deg";
            Cin>> recst.d3;
      recst .Av=(recst.d1+recst.d2+recst.d3)/3.0;
      Cout<<"Av="<< recst .Av;
}
```

Ex: writ a program to find average of n students has every one (8) degrees, using struct?

Sol:

```
# include <iostrem.h>
Struct student
{
        char name[10];
        int d[8] ;
        float AV;
};
main (  )
{
        Student A[n]; int n, i, j;
        cout <<" enter n"; cin>>n;
        for (i=1;i<=n ;i++)
        {
                cout<<"enter the name";
                cin>>A[i]. name
                Sum=0
                for (j=1;j<=8  ; j++)
                {
                        Cout<<"enter deg ";
                        sin>> A[i].d[j];
                        sum= sum + A[i] .d[j];
                }
                A[i].Av= sum / 8.0;
        }
        for ( i=1 ; i<= n ; i++)
        cout<<A[i].Av;
}
```