بسم الله الرحمن الرحيم

```
def add5(x):
    return x+5

def dotwrite(ast):
    nodename = getNodename()
    label=symbol.sym_name.get(int(ast[0]),ast[0])
    print '    %s [label="%s' % (nodename, label),
    if isinstance(ast[1], str):
        print ' %s' % ast[1]
        else:
            print '"];'
    else:
        print '"];'
        children = []
        for  in n, childenumerate(ast[1:]):
            children.append(dotwrite(child))
        print ,'    %s -> {' % nodename
        for  in :namechildren
            print '%s' % name,
```

CSE131: Computer Programming

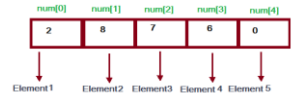
Lecture (3)

﴿وَقل رَّبِّ زِدْنِي عِلْماً﴾

(طه: ١١٤)

Array: data structures that allow us to store data of the same type in contiguous memory locations

⇒ they are a consecutive group of variables, all those constituting variables have the same data type & share a common name.



*1D Array

*array name is it's address

- declaration of array ⇒ `int array_name[array_size];`
  - array_name: اسمه هنا
  - array_size: عدد الـ elements اللي فيه
- initialization of array
  - ① `int a[3] = {1,2,3} ;` (a[ ] ممكن اسيبها فاضية)
  - ②
    ```
    int a[3];
    a[0] = {1};
    a[1] = {2};
    a[2] = {3};
    ```

Syntax error
- لو أعرف عدد elements أكتر من اللي محدده لل array هيعمل error
- هنا حددت تلاتة وعرفت اربعة

لو حالة العكس حدث اربعة وعرفت قيم أقل من اربعة الباقية هيكونوا بصفر تلقائي

- Size of operator: sizeof(…)
  - if used with data types, it returns the amount of memory is allocated to that data type.
  - ممكن استخدمه لو مش عارف عدد الـ elements بتاعة الـ array

```
n = sizeof(degree)/sizeof(used_data_type)
```

- to print elements in the array:

```
int x[] = {1,2,3,4,5};
for(int i=0;i<5;i++)
    printf("x[%d] = %d\n",i,x[i]);
```

*2D Array

- declaration of array ⇒ `int array_name[rows][columns];`
  - array_name: اسمه هنا
  - rows: عدد الصفوف
  - columns: عدد الأعمدة
- initialization of array:
  - ① `int a[3][3] = {1,2,3,4,5,6,7,8,9};` (لا جربنا الطريقة دي)
  - ② `int a[3][3] = {` (a[ ][3] ممكن اسيبها فاضية / a[3][ ] ميصحش اسيبها فاضية)
    ```
    {1,2,3}
    {4,5,6}
    {7,8,9}
    };
    ```
    brackets for rows
    brackets for array لا

	Column 0	Column 1	Column 2
Row 0	x[0][0]	x[0][1]	x[0][2]
Row 1	x[1][0]	x[1][1]	x[1][2]
Row 2	x[2][0]	x[2][1]	x[2][2]

* خد بالك:
→ a[2][5] - the 3rd value (row) in 6th column

- to print elements in the array:

```
int a[3][3] = {1,2,3,4,5,6,7,8,9};
 for(int i=0;i<3;i++)
{
        for(int j=0;j<3;j++)
                printf("a[%d][%d] = %d\t",i,j,a[i][j]);
        printf("\n");
}
```

*Strings:

- initialization of array:
  - ① `char text[] = {'H','e','l','l','o', 0};` (terminator: بتعرف البرنامج إن دي آخر الـ string اللازم تتكتب هنا ('\0'))
  - ② `char text[] = {"Hello"};`
  - ③ `char text[] = "Hello";`

- if the string is in form of multid array ⇒ (نفس طريقة الـ array 2D)

```
char greetings[3][10] = {
                        "hello",
                        "world",
                        "!"
                        };
```

- to print string text:

```
① char str[20];
  gets(str);
  puts(str);   = printf("%s", str);

② int main()
  {
  char greetings[3][10] = {
                          "hello",
                          "world",
                          "!"
                          };

  for(int i=0;i<3;i++)
        printf("%s ",greetings[i]);

  }
```

to print text

- taking String input:

```
① char str[20];
  gets(str);          // بتاخد كله
  printf("%s", str);

② char str[20];       // مش بتخد & مع الـ array
  scanf("%s", str);
  printf("%s", str);
```

بتاخد لحد اول مسافة فقط

*function & purpose:

```
#include <string.h>
```

① `strcpy(s1,s2);` Copies one String into another

② `strcat(s1,s2);` it's used to Combine two strings (بتلزق الـ String التاني في الأول)

③ `strlen(s1);` it's used to show the length of a string (عدد الحروف بس)

④ `strcmp(s1,s2);` it's used to compare two strings

`strcmpi(s1,s2);` بتتجاهل الـ upper case و الـ lower case مع بعض في المقارنة *

⇒ Comparing has 3 Cases: 0 → same; 1, -1 → different

⑤ `strrev(s1);` it's used to show the reverse of a string

⑥ `strlwr(s1);` Change string case to lower

`strupr(s1);` Change string case to upper

used in pointers • Returns a pointer to the first occurrence of ....

⑦ `strchr(s1,c);` character c in string s1

`strstr(s1,s2);` String s2 in string S1

```
#include <stdlib.h>
```

`x = atoi(…);` Convert string to integer value