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):
        if ast[1].strip():
            print '=%s' % ast[1]
        else:
            print '"];'
    else:
        print '"];'
        children = []
        for n, child in enumerate(ast[1:]):
            children.append(dotwrite(child))
        print ,'    %s -> {' % nodename
        for in :namechildren
            print '%s' % name,
```

CSE131: Computer Programming

Chapter (1&2)

﴿وَقل رَّبِّ زِدْنِي عِلْماً﴾

(طه: ١١٤)

Ahmed Juba

*Compiler : changes code into machine language

the first function excuted in the program is the main function

الكود بيشتغل بالترتيب من فوق لتحت

it's the entry point of the code

```
#include "stdio.h"
```
→ header → Reference from which we write orders [libraries]

main function

```
void main()
{
    printf("Hello in England\r\n");
}
```

main function body

function name; start bracket; End bracket; String Start; input Parameters (String); String end; \r\n = end line

1] printf("...\r\n"); → Printing function

2] int ... = ... ; (integer; Parameter name; integer number) → Calculate → we may not write them as we wish to use it later

```
printf("..%d\r\n",x);
```

numerical Variable — identifying this Variable respectivily → Variable name convection

if we wish to write in float
1] float ... = 2.506f → decrease .exe file size
2] %f

Variable name convection:
- up to 256 character
- don't use space, use "_"
- don't use function names
- don't start with no.

3] scanf("%d", &x); → interaction with the user

4] //place a comment

```
/*place
 *a comment*/
```

not essential — you can write multiple lines

→ place a comment to declare the code's function

Data types

→ Computer Supports numerical Values only

→ Available Values types
- integer Values
  - char
  - Short
  - int
    - int → Machine dependent (processor 32 bit or 64 bit)
    - long → Constant
- Real Values (rational)
  - Float (%f)
  - double (%d)

Development enviroment (البرنامج)

→ Compiler program Files → Binaries (*.obj) [Syntax error] → Linker binaries (*.obj) → Executable (*.exe)

Syntax error: error in writing code functions & strings

Run time error: error found after running the program like errors in string

Debug: checking the correction of the code
- Start debugging
- break point
- Step over

Special characters (*till now)
- ""
- ;
- &
- %
- \

inserting a character

1] `char ... = ... ;`

- direct letter → write between quotation mark
  - Ex: `char L1 = 'H' ;`
- decimal ASCII Code
  - Ex: `char L2 = 108 ;`
- hexdecimal ASCII Code
  - Ex: `char L3 = 0x6f ;`

to print the character

```
printf("%c", L1);
```

Formatting Examples:

1] \r\n → makes new line

2] \t → tap "Space of 8 letter"

3] \\ → Prints backslash

4] \" → prints double quotation

5] %x → to print Small letter → to assign as hexdecimal int

6] %X → to print Capital letter

7] %o → to assign as octal int

8] %d → to assign as decimal int

9] %c → to assign as character

10] %f → to assign as Float

11] %lf → to assign as double

12] %u → to assign as Unsigned int

13] %10f → to Fix space for digits (for digits appended by system)

type casting:

use when you are determined to do a certain conversion

Ex: `a = (int)b , b = (char)c`

Logic Expression:

1] OR → `Z = X|Y`

2] AND → `Z = X&Y`

3] XOR → `Z = X^Y`

4] NOT → `Z = ~X`

5] Shift
- Right → `X = X >> 2` (2 = no. of digits Shifted)
- left → `X = X << 2`

Problem → if you want change certain digits in binary integar

(change: set / reset)

Solution → Use logical operation AND OR

as it's faster and simple

```
#include "stdio.h"

void main()                                   // 'main' must return 'int'
{
    int x = 0xA7;
    int n;
    int y;

    printf("Enter the bit position:");
    scanf("%d", &n);                          // 'scanf' is deprecated: This function or variable may be unsaf...

    // Find a way to set the bit number n to 1 and print the result
    x = x | (1<<n);

    printf("y = %X\n", y);

    // Find a way to reset the bit number n to 0 and print the result
    x = x & ~(1<<n);

    printf("y = %X\n", y);
}
```

#define ... ...

used to define certain constant or variable or message ...etc

- string replace
- constant (π, e)
  - `#define PI 3.141592`

#include "math.h"

Mathematical Operation Let x = 2.1 and z = 1.2	C Example
$y = \sin(x)$	`y = sin(x);`
$y = \cos(x)$	`y = cos(x);`
$y = \tan(x)$	`y = tan(x);`
$y = \sin^{-1}(x)$	`y = asin(x);`
$y = \cos^{-1}(x)$	`y = acos(x);`
$y = \tan^{-1}(x)$	`y = atan(x);`
$y = \tan^{-1}(x/z)$	`y = atan2(x, z);` `//to avoid dividing by 0 if b = 0`
$y = \sqrt{x}$	`y = sqrt(x);`
$y = (x)^{3.22}$	`y = pow(x, 3.22);`
$y = ln(x)$	`y = log(x);`
$y = log_{10}(x)$	`y = log10(x);`

$y = \|x\| = 2.1$	`y = abs(x); //if x is integer`
	`y = fabs(x); //if x is real`

$y = \lceil x \rceil = 3$	`y = ceil(x);` → next number
$y = \lfloor x \rfloor = 2$	`y = floor(x);` → Previous number

Mathematical Expressions:

C Expression	Meaning
X = X + 9;	Calculate X+9 then stores the result in X
X++;	Add one to X
X--;	Subtract one from X
X = 10; Y = 5; X = X + Y++;	Add X+Y → 15 then increment Y → 6 Store 15 in X
X = 10; Y = 5; X = X + ++Y;	Increment Y → 6 Add X+Y → 16 Store 16 in X
X = 5; X += 6;	Add 6 to X → X = 11
X -= 1;	Subtract 1 from X → X = 10
X *= 2;	Multiply X by 2 → X = 20
X /= 5;	Divide X by 5 → X = 4

Steps of operation:

`float Z = X/Y`

1] Expression Calculation

2] Allocation

3] Assignment

*Program Execution Flow:

Decision: if, switch
Loop: for, while, do while
Direct Jump: goto

1] Conditions:

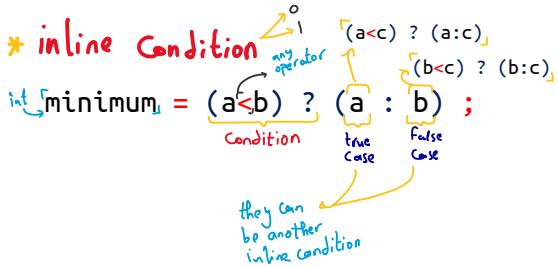
Operator	Meaning
>	Greater
>=	Greater or equal
<	Less
<=	Less than or equal
==	Equal
!=	Not equal
!	Not If the input is true the output is false if the input is false the output is true
&&	And Example: A>B && C>D If both sides are true the output is true, otherwise it gives false
||	Or Example: A>B || C>D If wither sides is true the output is true, otherwise it gives false

⇒ Logical Operation:

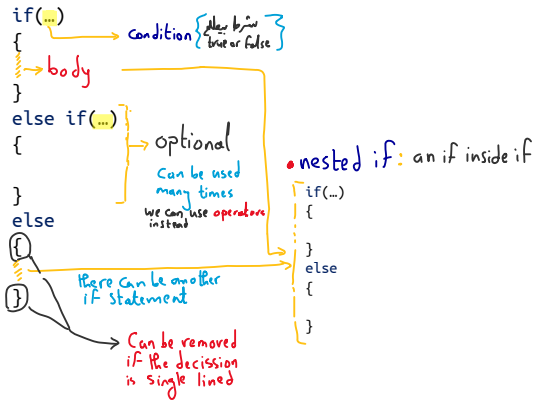
1) Bitwise: acts on bits (| & ^ ~)

2) Conditional acts on whole Variable (|| && !)
- وجود رقم → true
- صفر → false

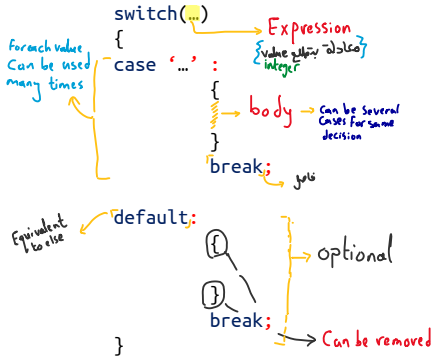
* inline Condition



1) if statement

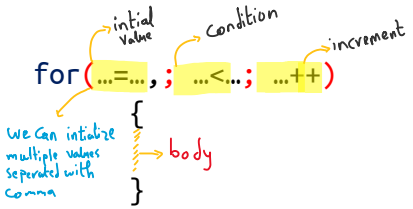


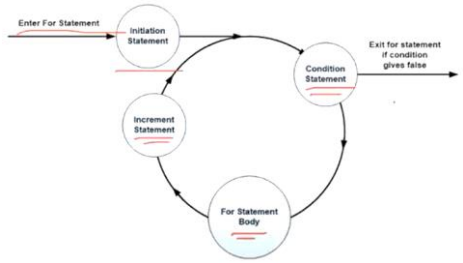
2) switch statement



2] loops:

1) for:



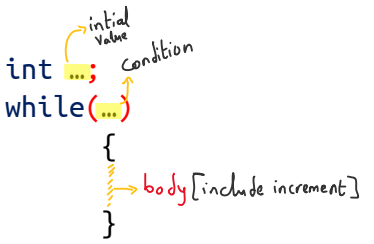



```
for(;;)
{
}
```
Cycling Forever

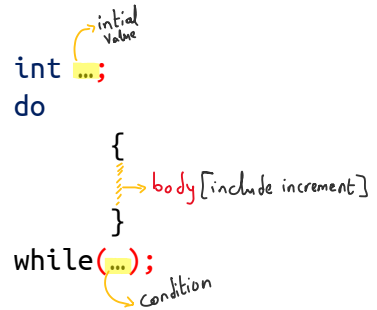
```
int n = 0; //initiation

for(;;) {
    printf("%d: Hello World\n", n);
    n++; //Increment
    if(n>10) break; //Condition
}
```

2) while:



3) do while:




```
• #include "conio.h"
choice = getche() ;
if (choice=='...')
```
- Echo: to show the typed character (Can be removed)
- switch (getche()) — Labels
- 'Single character'

3] goto → basic statement of loops