



C Programming

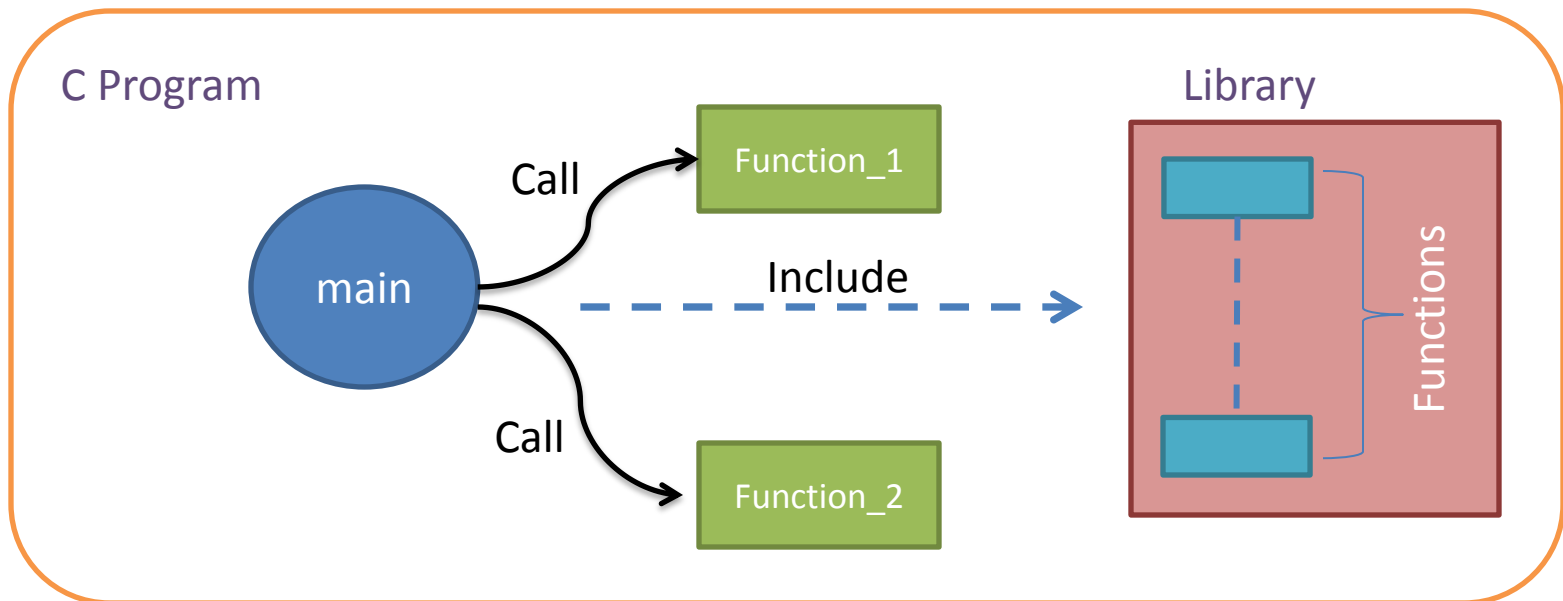
Lecture One

C Programming Basics

*This material is developed by IMTSchool for educational use only
All copyrights are reserved*

Introduction To C Programming

- C is structured programming, it means that the C program is composed of small parts each part called "function".
- The first function to be executed (The entry point of the program) is called "main".
- Sometimes, we may write some functions in a stand alone file for organizing, this file is called "library".



Hello World in C

```
/* Include stdio.h library
   To use printf function */
#include <stdio.h>

// define the main function
void main(void)
{
    /* Call the main function and
       pass string to it to print */
    printf("Hello C world");
}
```

Multiline comment

Include command

Single line comment

printf function call

Any line inside a function must ends with semicolon ;

Comments in C

Comments are non-executable text used to provide documentation for the code. It provide clarity to the C source code allowing others to better understand what the code was intended to accomplish.

It is always recommended to use comments in your code, for that in IMT we have a rule, at least one comment for each code line.

1- Single-line comment

Any line preceded by two forward slashes `//`.

```
// This is single line comment
```

2- Multi-line comment

Any text starts with `/*` and ends with `*/`

```
/* This is multiline comment */
```

Strings in C

- It is comprised of a set of characters that can also contain spaces, special characters and numbers.
- In C string is represented between double quotation " *This is string* ".
- `printf` function will print the string passed to it as it is.
- Escape operator may be used inside the string `\`, this operator may insert tab, new line or quotes.



```
printf("My Name is Ahmed\n");  
printf("I'm 26 Years Old  ");
```

<code>\n</code>	New line
<code>\t</code>	Horizontal tab
<code>\v</code>	Vertical tab
<code>\'</code>	Single quote
<code>\"</code>	Double quote

Write a C code that will print your short biography.

Full Name, Birth Year, Faculty, and graduation year

Expected Output

```
I'm Ahmed Assaf
```

```
My birth date is 22 Sep 1991
```

```
I graduated from Faculty of Engineering
```

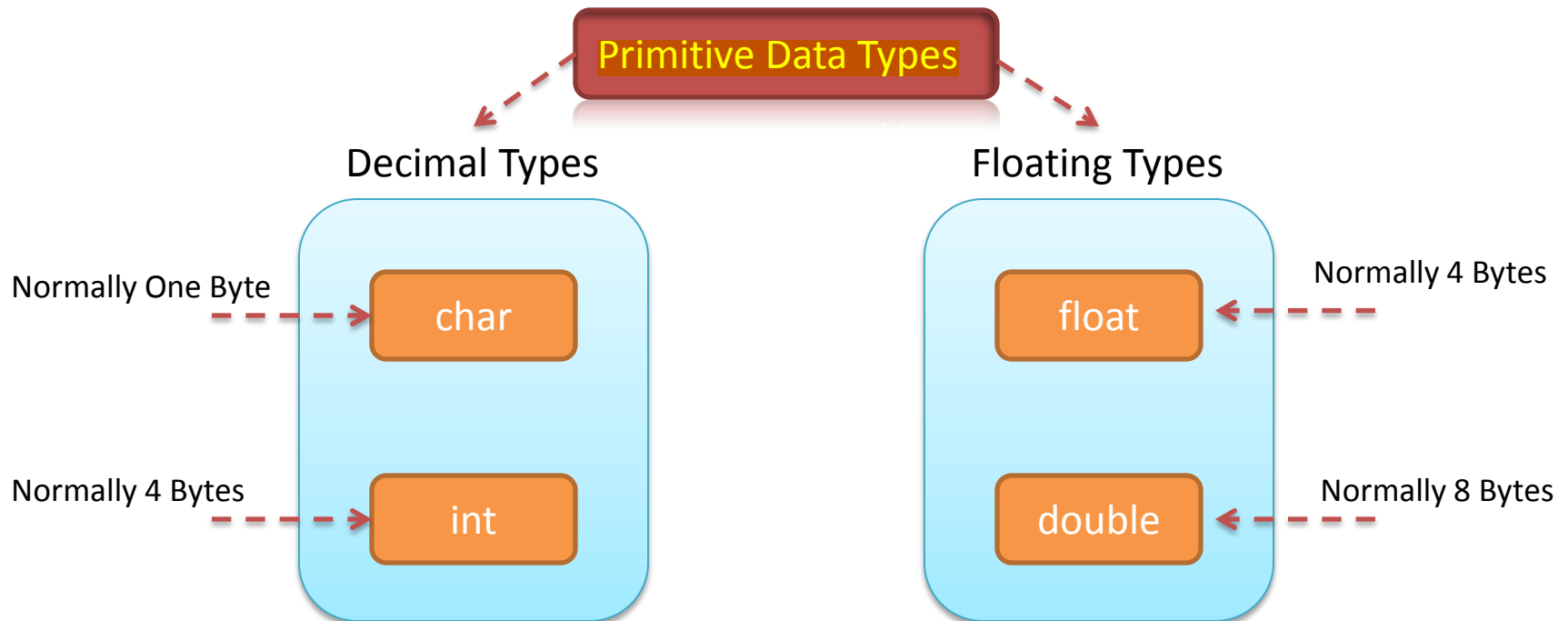
```
Cairo University 2013
```

Time To Code



variables in C

- Variable is a part from the memory, used to hold a piece of data.
- The variable has a *type*, *name* and *value*.
- The types of the variables differs in *Size* and/or *Data to be saved*.



Note, Data types size may differ from one compiler to other, this issue will be discussed later.

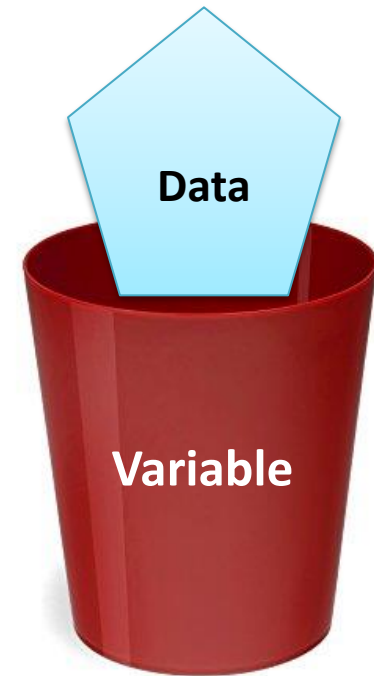
variables in C

Syntax

type name ;

Or

type name = initial_value ;



```
char x = 9;
```

Define char variable and initialize it with 9

```
int y;
```

Define int variable with initialization, it will have a random value, called **garbage**

```
float z = 6.52;
```

Define float variable and initialize it with floating number

Variable Naming Rules

1- Variable can contain:

- Capital Letters **A to Z**
- Small Letters **a to z**
- Numbers **0 to 9**
- Underscore **_**

2- First Character must be alphabet or Underscore

```
int a1;
```

← ----- Allowed

```
int 1a;
```

← ----- Not Allowed

3- Blanks & Commas are not allowed

4- No Special Symbols other than underscore are allowed, *ex ?, #, etc*

5- Variable name Should not be Reserved Word

6- Variable name can not be repeated in the same scope *"Will be clarified later"*

Question

What will be the output of the following code ... ?

```
#include <stdio.h>

void main(void)
{
    /* Define int variable and initialize 10 */
    int x = 10;

    printf ("The variable value is x");

}
```



Solution

```
#include <stdio.h>

void main(void)
{
    /* Define int variable and initialize 10 */
    int x = 10;

    printf ("The variable value is x");

}
```

Output

---> The variable value is x

The string will be printed as it is, it will not replace `x` with its value. Instead it will deal with `x` as a normal character not a variable.

Printing a variable

- **printf** function can print a variable inside the string, it could be done by inserting format specifier that will be replaced by the values specified in subsequent additional arguments.
- Example

```
printf ("The variable value is %d", x);
```



The output

```
The variable value is 10
```

This format specifier will be replaced by the value of x

Common specifiers used
with **printf** function



%d	Format specifier for decimal value
%f	Format specifier for floating value
%c	Format specifier for character value

Scanning a value

scanf function is a part from the ***stdio*** library, it is used to get value from the user and save it in a variable.

Syntax

`scanf("formatSpecifier" , &VariableName);`

This operator must be written
and will be discussed later



Example

```
/* Define a variable to save a value from user */  
int x;  
  
/* Get the value from the user */  
scanf("%d",&x);
```

Write a C code that will ask the user to enter a value then print it.

Expected Output

```
Please Enter the value: 10  
The value you entered is 10
```

Time To Code



C Operators

Arithmetic	Uni		++			--		
	Bi		+	-	*	/	%	
Bit wise	&		~	^	>>	<<		
Assignment	=	+=	-=	*=	/=	%=	+=	
	&=	=	^=	>>=	<<=			
Relational	>	<	>=	<=	==	!=		
Logical	&&					!		
Other	Size of operator			sizeof()				
	Ternary operator			? : ;				
	Address operator			& (will be discussed later)				
	Dereference			* (will be discussed later)				
	Subscriptor			[] (will be discussed later)				

Arithmetic Operators

```
int x = 10;
```

```
int y = 5 ;
```

Bi Operators, operators that takes two operands

1- Summation

example

```
int sum = x + y;    /* sum = 15 */
```

2- Subtraction

example

```
int sub = x - y;    /* sub = 10 */
```

3- Multiplication

example

```
int mul = x * y;    /* mul = 150 */
```

4- Division

example

```
int div = x / y;    /* div = 2 */
```


Arithmetic Operators

Bi Operators, operators that takes two operands

5- Modulus (reminder)

example

```
int mod = x % y;    /* mod = 0    */
```

example

```
int mod = y % x;    /* mod = 5    */
```

example

```
int mod = 10 % 3;    /* mod = 1    */
```

example

```
int mod = 9 % 1 ;    /* mod = 0    */
```

example

```
int mod = 17 / 9;    /* mod = 8    */
```

```
int x = 10;
```

```
int y = 5 ;
```

LAB 3

Expected Output

Write a C code that will ask the user to enter two values and print their summation and multiplication.

```
Please Enter number 1 : 10  
Please Enter number 2 : 20  
a + b = 30  
a x b = 200
```

Time To Code



Arithmetic Operators

Uni Operators, operators that takes one operand

```
int x = 10;
```

1- Increment

example

```
x++;
```

```
/* x = 11 */ ←----- Postfix
```

example

```
++x;
```

```
/* x = 11 */ ←----- Prefix
```

2- Decrement

example

```
x--;
```

```
/* x = 9 */ ←----- Postfix
```

example

```
--x;
```

```
/* x = 9 */ ←----- Prefix
```

Note: In previous examples, no difference between postfix and prefix cases

Arithmetic Operators

Uni Operators, operators that takes one operand

```
int x = 10;  
int y;
```

1- Increment

example

```
y = x++;      /* x = 11, y = 10 */
```

← Assign x to y, then increment x

example

```
y = ++x;      /* x = 11, y = 11 */
```

← Increment x, then assign x to y

2- Decrement

example

```
y = x--;      /* x = 9 , y = 10 */
```

← Assign x to y, then decrement x

example

```
y = --x;      /* x = 9 , y = 9 */
```

← Decrement x, then assign x to y

Bitwise Operators

To apply these operators correctly, let's first imagine these numbers in binary

x = 1010

y = 0101

```
int x = 10;  
int y = 5;
```

1- And

example `int and = x & y; /* and = 0 */`

2- Or

example `int or = x | y; /* or = 15 */`

3- Not

example

```
char not = ~x;      /* not = 11110101 in binary 245 decimal */
```

4- XOR

example `int xor = x ^ y; /* xor = 15 */`

Bitwise Operators

To apply these operators correctly, let's first imagine these numbers in binary

x = 1010

y = 0101

```
int x = 10;
int y = 5;
```

6- Right shift

example

Variable to shift

Shifting steps

```
int Right_shift = x >> 2;    /* Right_shift = 2 */
```

7- Left shift

example

```
int Left_shift = y << 2;    /* Right_shift = 20 */
```

= 010100 = 20

16 4

LAB 4

Solve these examples in a paper and confirm that your answers are correct by writing a code printing the result

```
x = 7;  
y = 4;
```

```
z = x & y;
```

```
k = x | y;
```

```
m = x ^ y;
```

```
L = x >> 1;
```

```
N = y << 2;
```

Time To Code



Assignment operators

1- Assign

example:

```
x = 20;    /* Assign 20 to x */
```

```
int x = 10;
```

2- Add and Assign

example:

```
x += 3;    /* Add 3 to x and assign the value to x, x = 13 */
```

3- Subtract and Assign

example:

```
x -= 4;    /* Sub 4 from x and assign the value to x, x = 6 */
```

4- Multiply and Assign

example:

```
x *= 5;    /* Multiply x by 5 and assign the value to x, x = 50 */
```

5- Divide and Assign

example:

```
x /= 2;    /* Divide x by 2 and assign the value to x, x = 5 */
```


Assignment operators

```
int x = 10;
```

6- Modulus and Assignment

example:

```
x %= 4;    /* Get the remainder of x divided by 4 and assign  
            the value to x, x = 2 */
```

7- And then Assign

example:

```
x &= 1;    /* Apply and operation between x and 1 and assign  
            the value to x, x = 0 */
```

8- Or then Assign

example:

```
x |= 15;   /* Apply or operation between x and 15 and assign  
            the value to x, x = 15 */
```

9- XOR then Assign

example:

```
x ^= 2;    /* Apply xor operation between x and 2 and assign  
            the value to x, x = 8 */
```

Assignment operators

10- Shift right then Assign

example:

shift right >> meaning that divide
by 2 every one shift (/2)

```
int x = 10;
```

x >>= 1; /* Apply right shift to x by 1 step and assign
the value to x, x = 5 */

11- Shift left then Assign

example:

shift left << meaning that multiply
by 2 every one shift (* 2)

x <<= 1; /* Apply left shift to x by 1 step and assign
the value to x, x = 20



The End ...

Other types of operators will be discussed later on,

*Any
questions
... ?*



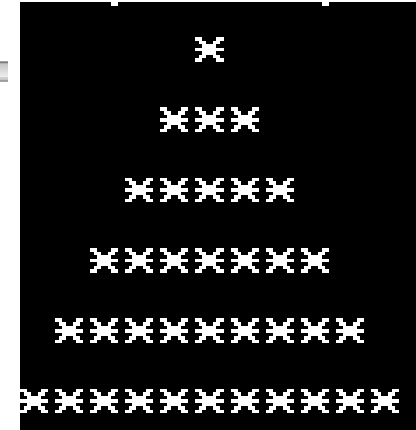
Assignment 1

Write a code that can draw this pyramid

```

1 //relation between no of rows and stars => (i*2-1)
2 //relation between no of rows and spaces => (rows-i)
3
4
5
6 #include <stdio.h>
7 #include <stdlib.h>
8
9 int main()
10 {
11     int row,i,x;
12     while(1)
13     {
14         printf("Enter no of rows: ");
15         scanf("%d",&row);
16         for(i=1;i<=row;i++)
17         {
18             for(x=0;x<row-i;x++) //spaces
19             {
20                 printf(" ");
21             }
22             for(x=0;x<1*2-1;x++) //stars
23             {
24                 printf("*");
25             }
26             printf("\n");
27         }
28         return 0;
29     }
30 }

```



Assignment

Assignment 2

Write a code that scan 3 numbers from the user and print them in reversed order

```
Please enter number 1: 11  
Please enter number 2: 12  
Please enter number 3: 13  
number 3: 13  
number 2: 12  
number 1: 11
```



Assignment 3

Write a code that takes 2 numbers and print their summation, subtraction, anding, oring, and Xoring

```
Please Enter number a: 4
Pleae Enter number b: 2
a + b = 6
a - b = 2
a & b = 0
a | b = 6
a ^ b = 6
```





www.imtschool.com



www.facebook.com/imaketechologyschool/

*This material is developed by IMTSchool for educational use only
All copyrights are reserved*