



Introduction to Programming

Lecture 2: Introduction to C

Mahmoud El-Gayyar

elgayyar@ci.suez.edu.eg



Review Lecture 1

- *Introduction to the course*
 - ◆ General information
 - ◆ Syllabus
 - ◆ Course organization
 - ◆ General rules
- *Introduction to Programming*
 - ◆ Programming Skills
 - ◆ Programming Model
 - ◆ Elements of a real Programming Languages
 - ◆ History of C
 - ◆ Compiler Terminology

Review Lecture 1

- *Introduction to the course*
 - ◆ General information
 - ◆ Syllabus
 - ◆ Course organization
 - ◆ General rules
- *Introduction to Programming*
 - ◆ Programming Skills
 - ◆ Programming Model
 - ◆ Elements of a real Programming Languages
 - ◆ History of C
 - ◆ Compiler Terminology

Outline

- *Introduction to C Language*
- *Your First C Program*
- *Let's have another example*
- *Program Structure*

Outline

- *Introduction to C Language*
- *Your First C Program*
- *Let's have another example*
- *Program Structure*

Programming Language

- *A programming language is a tool*
 - ◆ Can not perform every task unaided
- *C language has built-in functions*
 - ◆ But not for every thing
- *C is a dangerous language*
 - ◆ Provide low level access to memory
 - ◆ You should be careful while dealing with it



Outline

- *Introduction to C Language*
- ***Your First C Program***
- *Let's have another example*
- *Program Structure*

Outline

- *Introduction to C Language*
- ***Your First C Program***
- *Let's have another example*
- *Program Structure*

Your First C Program

```
#include <stdio.h>

int main()
{

    printf("Hello, world!\n");
    return 0;

}
```

Your First C Program

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    printf("Hello, world!\n");
```

```
    return 0;
```

```
}
```

Your First C Program

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    printf("Hello, world!\n");
```

```
    return 0;
```

```
}
```

Your First C Program

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
printf("Hello, world!\n");
```

```
return 0;
```

```
}
```

Your First C Program

```
#include <stdio.h>

int main()
{

    printf("Hello, world!\n");
    return 0;

}
```

Outline

- *Introduction to C Language*
- *Your First C Program*
- ***Let's have another example***
- *Program Structure*

Your Second C Program

```
#include <stdio.h>
/* print a few numbers, to illustrate a
simple loop */
int main()
{
    int i; // single line comment
    for(i = 0; i < 10; i = i + 1)
        printf("i is %d\n", i);
    return 0;
}
```

Your Second C Program

```
#include <stdio.h>
/* print a few numbers, to illustrate a
simple loop */
int main()
{
    int i; // single line comment
    for(i = 0; i < 10; i = i + 1)
        printf("i is %d\n", i);
    return 0;
}
```


Your Second C Program

```
#include <stdio.h>
/* print a few numbers, to illustrate a
simple loop */
int main()
{
    int i; // single line comment
    for(i = 0; i < 10; i = i + 1)
        printf("i is %d\n", i);
    return 0;
}
```

Your Second C Program

```
#include <stdio.h>
/* print a few numbers, to illustrate a
simple loop */
int main()
{
    int i; // single line comment
    for(i = 0; i < 10; i = i + 1)
        printf("i is %d\n", i);
    return 0;
}
```

Outline

- *Introduction to C Language*
- *Your First C Program*
- *Let's have another example*
- ***Program Structure***

Program Structure

A number is automatically assigned to each line. You *don't* have to type these.

Line 1 includes instructions from another code file, called `machinescience.h`. Your code files should *always* begin with this line.

Line 2 marks the start of your main function. Every C program *must* have a function called "main."

```
1 #include <machinescience.h>
2 int main(void)
3 {
4     INSTRUCTIONS GO HERE    //This is a comment
5 }
```

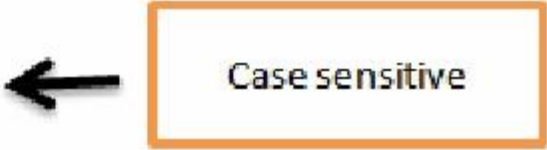
The main function *always* begins and ends with curly braces: { and }.

Your program instructions are contained in the curly braces of the main function.

Text preceded by two slashes is called a comment. Comments are ignored by the compiler. Usually, they describe what each line of code does.

C is Case-Sensitive

Unique distinct values
AA
aa
aA
BB
bb



The diagram illustrates the concept of case sensitivity in C. It features a table with the title "Unique distinct values" containing six rows: "AA", "aa", "aA", "BB", "bb", and three empty rows. To the right of the table is a box with an orange border containing the text "Case sensitive". A black arrow points from this box to the table, indicating that the values in the table are treated as distinct due to case sensitivity.

- So if you write ***Main*** instead of ***main*** or ***iNt*** instead of ***int***, the compiler will give you an error

1st Good Programming Style (Indentation)

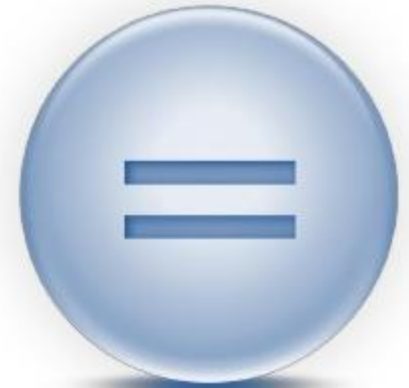
- *C don't care about spaces*

```
for(i = 0; i < 10; i = i + 1)
    printf("%d\n", i);
```

```
for(i = 0; i < 10; i = i + 1) printf("%d\n", i);
```

```
    for(i = 0; i < 10; i = i + 1)
printf("%d\n", i);
```

```
for      (      i
=        0      ;
i        <      10
;        i      =
i        +      1
)        printf (
"%d\n"   ,      i
)        ;
```



Summary

- *C program*
 - ◆ May contain several functions
 - ◆ It can also use built-in functions (don't forget the include)
 - ◆ Must have a main function
 - ▶ *It may return a value (0, -1)*
 - ◆ Comments can be used to enrich your program and make it more readable
- *C is case sensitive*
- *Indentations (white spaces) is a good programming style*