Introduction to Programming, Basic Structure of C Programs

ESC101: Fundamentals of Computing

Nisheeth

Announcements

- Please make sure you know your section number for ESC101
  - Refer to the student list shared on the course website. Final list will be uploaded by Friday evening.
- Regularly visit the course website. Slides for each lecture (and other material) will be posted there. Slides in PPTX (Power-point) and PDF

- Please make sure you can access Piazza (and can get email notifications of the messages posted on Piazza in real-time or digest mode)

- Prutor availability
  - During lab hours (1400-1500, M/Tu/Wed/Thu), only in NCL labs
  - Outside lab hours, hostels, CC, NCL etc (NCL open till 2AM)
  - Correct Prutor link: https://esc101.cse.iitk.ac.in/ (NOT https://prutor.cse.iitk.ac.in/)

Announcements

- When logging in on the lab machines (Linux/Windows), use your CC id (without @iitk.ac.in) and your CC password
- When logging in on the Prutor website , use your CC id (with @iitk.ac.in) and your CC password
- Unable to access the course website and Prutor?
  - Are you using a data plan on your smart phone?
  - Our course website, Prutor are *internal*, not accessible outside IITK
    - Solution 1: use IITK computers (CC, NCL, hostel)
    - Solution 2: install a VPN app on your smart phone https://www.iitk.ac.in/ccnew/index.php/13-network/99-how-to-use-ssl-vpn
  - Piazza is accessible from all places

Announcements

- Hindi lecture videos of many topics in ESC101 are available online
  - https://onlinecourses.iitk.ac.in/esc101_hindi/ (created by Prof. Rajat Mittal and his team, link also under References on course website)

- We will soon hold some special sessions for students who do not feel very comfortable with English (will discuss what is being covered in lectures)
  - Will circulate a form to ask if you need it

- We will soon hold a special lab session for students who are not familiar with operating computers
  - Will circulate a form to ask if you need it

Programming: Some Benefits

- Applications in Engineering (civil, chemical), Sciences, Economics, AI
  https://www.youtube.com/watch?v=nKIu9yen5nc

- Even artists, comedians need to code ☺
  https://www.youtube.com/watch?v=EFwa5Owp0-k

- Be prepared for the future – job markets changing rapidly

- People who can code often deal with day-to-day problems more efficiently

How to Communicate with Computers?

- We need a language to communicate with the computer hardware

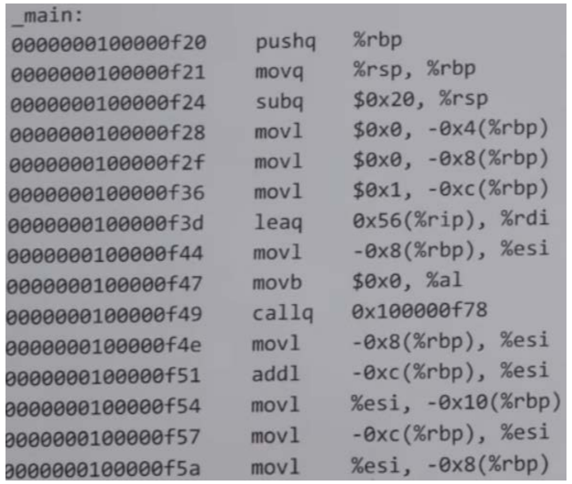
- The language should be one that the computer's hardware understands

Picture courtesy: www.professionaladviser.com/

How to Communicate with Computers?

- One way is by using the machine language that the hardware understands

- Every type of computer hardware has a specific machine language

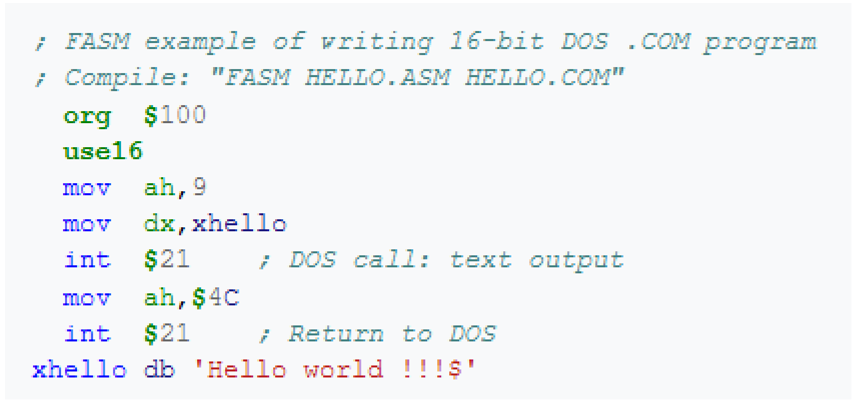


```
_main:
0000000100000f20    pushq   %rbp
0000000100000f21    movq    %rsp, %rbp
0000000100000f24    subq    $0x20, %rsp
0000000100000f28    movl    $0x0, -0x4(%rbp)
0000000100000f2f    movl    $0x0, -0x8(%rbp)
0000000100000f36    movl    $0x1, -0xc(%rbp)
0000000100000f3d    leaq    0x56(%rip), %rdi
0000000100000f44    movl    -0x8(%rbp), %esi
0000000100000f47    movb    $0x0, %al
0000000100000f49    callq   0x100000f78
0000000100000f4e    movl    -0x8(%rbp), %esi
0000000100000f51    addl    -0xc(%rbp), %esi
0000000100000f54    movl    %esi, -0x10(%rbp)
0000000100000f57    movl    -0xc(%rbp), %esi
0000000100000f5a    movl    %esi, -0x8(%rbp)
```

- However, using machine language is tedious/unnatural for humans

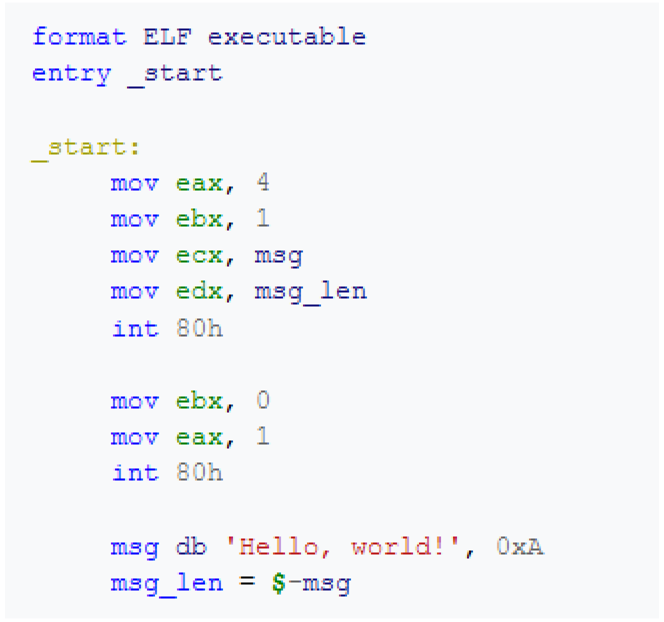
- Also need to re-write machine code if we want to run the code on another computer that has a different type of hardware – cumbersome

Hello World (in assembly language)



```
; FASM example of writing 16-bit DOS .COM program
; Compile: "FASM HELLO.ASM HELLO.COM"
  org  $100
  use16
  mov  ah,9
  mov  dx,xhello
  int  $21     ; DOS call: text output
  mov  ah,$4C
  int  $21     ; Return to DOS
xhello db 'Hello world !!!$'
```

Intel x86, DOS



```
format ELF executable
entry _start

_start:
     mov eax, 4
     mov ebx, 1
     mov ecx, msg
     mov edx, msg_len
     int 80h

     mov ebx, 0
     mov eax, 1
     int 80h

     msg db 'Hello, world!', 0xA
     msg_len = $-msg
```

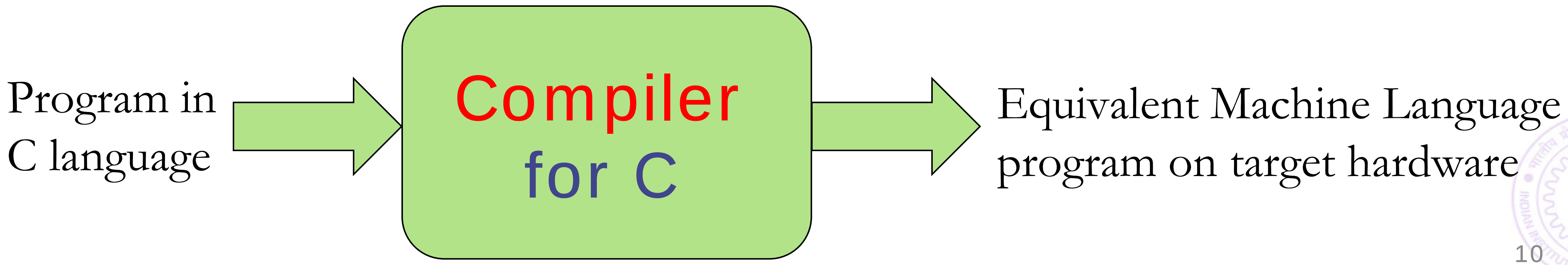
Intel x86, Linux

Computers and Programming

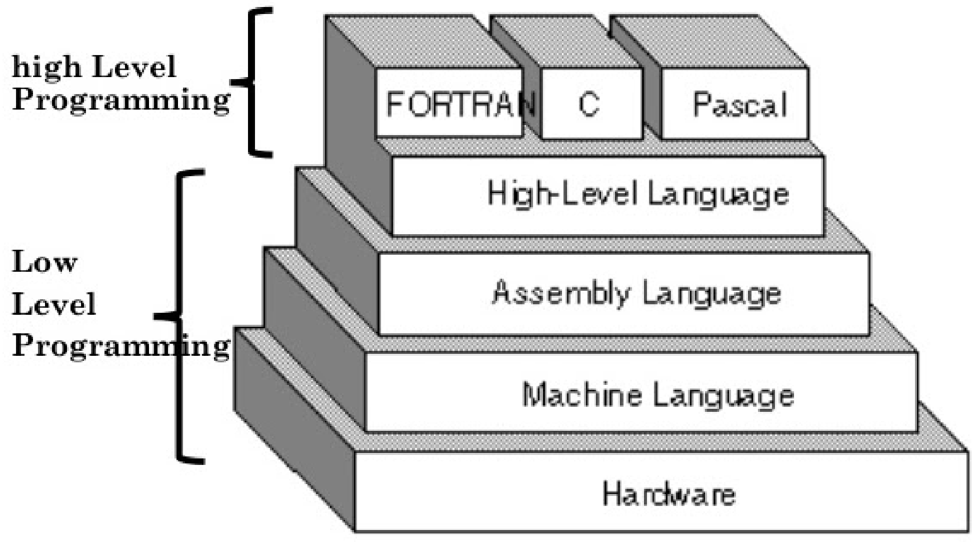
- A better alternative would be to write our programs in a language that is
  - Easy for us to write/understand
  - Easy to port it to different types of computer hardware without re-writing the code
- **High-level** programming languages like C make it possible
- How: Write the code in a high-level language and translate it into machine language using another software called "compiler"

Computers and Programming

- A better alternative would be to write our programs in a language that is
  - Easy for us to write/understand
  - Easy to port it to different types of computer hardware without re-writing the code
- **High-level** programming languages like C make it possible



Low-level vs High-level Languages



```
while(n>0)
{
sum = sum + n;
--n;
}
```

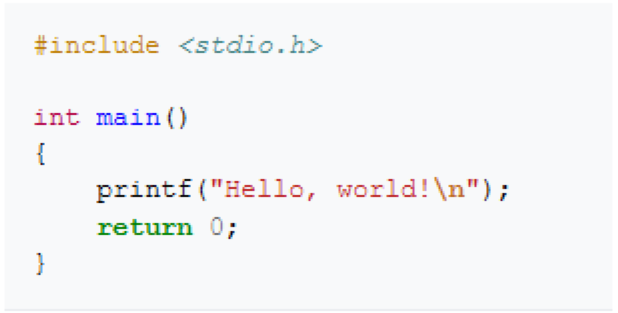
High-level
(example: C)

```
L28   movf    _n,f
      btfsc   STATUS,Z
       goto   L41
      movf    _n,f
      addwf   _sum,f
      btfsc   STATUS,C
       incf   _sum+1,f
      decf    _n,f
      goto    L28
L41
```

Low-level
(example: Assembly)

- Low-level: Form is closer to what the machine's hardware understands
  - Examples: Machine Language, Assembly Language
- High-level: Form is closer to what humans understand
  - Examples: C, C++, Python, etc

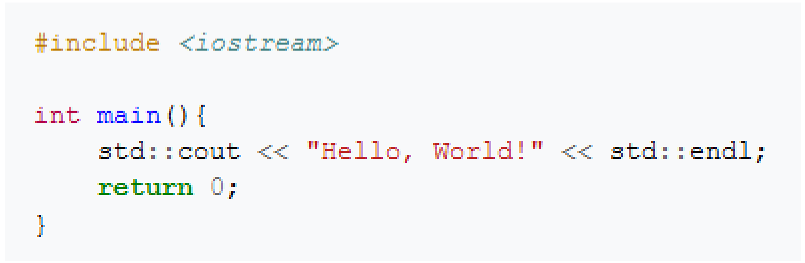
Hello World



```c
#include <stdio.h>

int main()
{
    printf("Hello, world!\n");
    return 0;
}
```

C

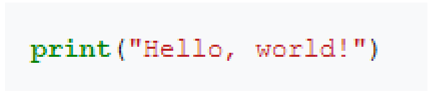


```cpp
#include <iostream>

int main(){
    std::cout << "Hello, World!" << std::endl;
    return 0;
}
```

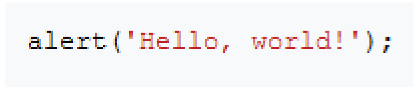
C++

Compiled languages



```python
print("Hello, world!")
```

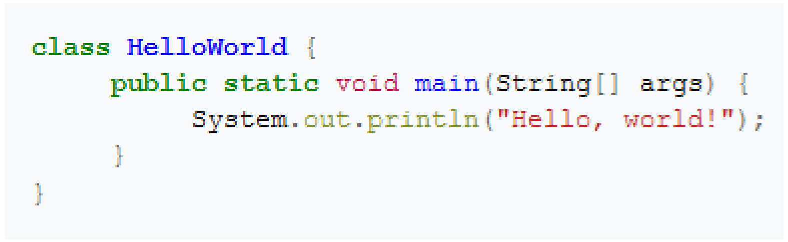
python



```javascript
alert('Hello, world!');
```

JavaScript

Interpreted languages



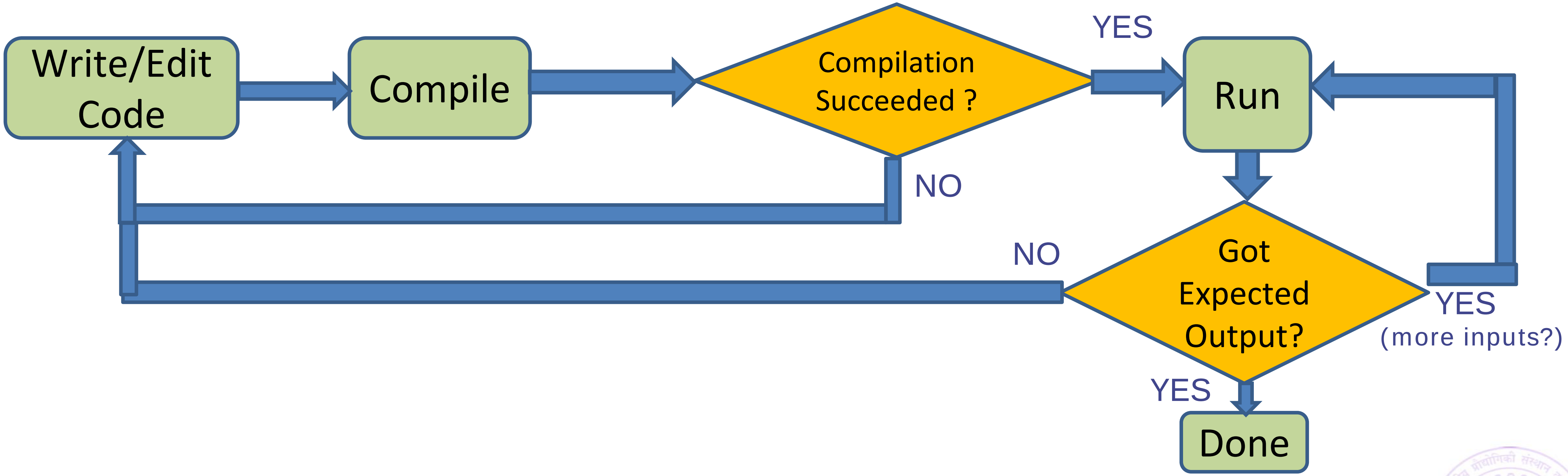
```java
class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, world!");
    }
}
```

Java

Runtime languages

Programming Cycle for Compiled Languages

(The typical cycle)



Note: Some high-level languages are not compiled but use an "interpreter" to communicate with the hardware (Example: Python, MATLAB, etc)

The C Programming Language

- A high-level programming language

- Originally developed by Dennis Ritchie (1972) to design the UNIX operating system and applications running on UNIX

- Widely used. Many operating systems, and even parts of many other programming languages such as Python were developed using C

- You are going to learn C language in this course
  - Be patient at the beginning
  - Some things may seem unfamiliar, strange for few days
  - Will get used to these very quickly
  - Best way to learn a new language – speak it and practice! (on Prutor and other places)

A Simple C Program

```
#include<stdio.h>
int main(){
    printf(“Welcome to ESC101”);
    return 0;
}
```

The program prints “Welcome to ESC101” (without the quotes)

Structure of A Simple C Program

Every C program's **entry point** (program's **execution** starts here) is the **main** function with **return type integer**

Tells C compiler to include the standard input/output library **stdio.h** (collection of **functions** such as printf, scanf, etc)

```
#include<stdio.h>
int main(){
    printf("Welcome to ESC101");
    return 0;
}
```

main function must open with left curly brace {

printf function prints a user specified output

Every statement in a C program must end with semi-colon ;

main function must close with right curly brace }

The main function must return an integer (return 0 means successful execution of program)

`printf("Welcome to ESC101")` and `return 0` are 'statements' in the above code. Each C statement must end with a semi-colon ;

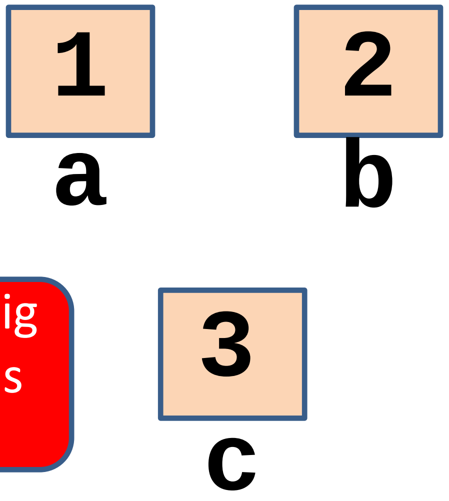
Another Simple C Program

Spaces are okay at some places

```
# include <stdio.h>
int main () {
    int a = 1;
    int b = 2;
    int c;
    c = a + b;
    printf(“Result is %d”, c);
    return 0;
}
```

Each **variable’s declaration** creates a “box” big enough to store it at a **location** in computer’s main memory (RAM)

Assigning a value to the variable writes that value in the box



= and + are “operators”

= is assignment operator

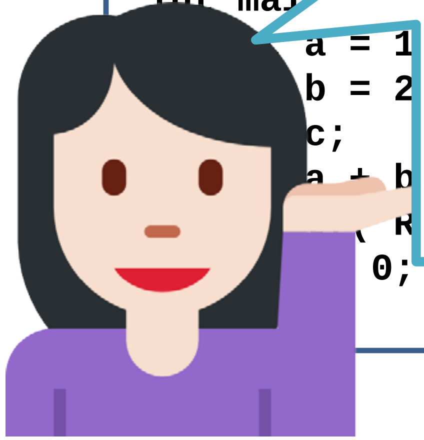
+ is addition operator

a+b is an “expression”

The program prints the message “Result is 3”

Multiple Ways of Writing Code: Same Effect

```
# include <
 int mai
     a = 1
     b = 2
     c;
     a + b
     ( R
     0;
```



Explore, practice. It will take only a few days to internalize.

```
tdio.h>
 {



b;
    printf(“Result i
    return 0;
}
```

Declare all then assign values

```
# include <stdio.h>
 int main ()  {
    int a = 1;
    int b = 2;
    int c = a + b;
    printf(“Result is %d”, c);
    return 0;
```

Shortcut

How will I remember all this?

```
# include <stdio.h>
 int main ()  {
    int a,b,c;
    a = 1;
    b = 2;
    c = a + b;
    printf(“Result is %d”, c);
    return 0;
 }
```

Shortcut

```
# include <stdio.h>
 int main ()  {
    int a=1,b=2,c;
    c = a + b;
    printf(“Result is %d”, c);
    return 0;
 }
```

Shortcut

And other possible ways too…

The ‘printf’ Function

- A function used for printing the outputs of the C program

- Prints the outputs in a format specified by us

- We have already seen some simple examples of usage of printf

```
printf(“Welcome to ESC101”);
```

%d means that we want to print the value of an **integer** variable (named c here)

```
printf(“Result is %d”, c);
```

More on printf in the next class…