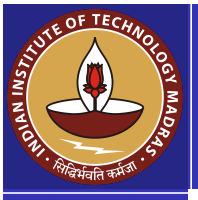



# C++ Programming Basics
# Procedural Aspects

Saurav Samantaray [1]

[1]Department of Mathematics
IIT Madras

August 1, 2024




# The Very First C++ Code



- Let the computer greet you.

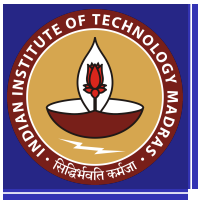
```
#include<iostream>
using namespace std;

// every program has a main
int main()
    {
    // print hello world and shift to
    // the next line
    cout << ‘‘Hello World’’ << endl;
    return 0;
    }
```

- Save the above into a file "hello.cpp".


# Compiling a C++ Code



- `g++ -c hello.cpp`.
- This only compiles the code and checks if all the syntaxes make sense or not.
- How do we run this?
- `g++ -o hello.exe hello.cpp`
- `./hello.exe`.




# Program To Illustrate Basic Features of C++



**Task** *Write a program that takes in two integers and as input and prints the sum of all integers between them.*

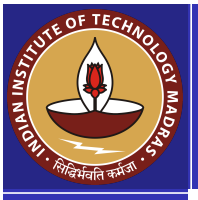
- It should be able to take in two integers, lets say "a" and "b".
- It should print the final sum.
- It should have a way to understand $a > b$ or vice-versa.


# Variable Declaration



```
int a, b;
```

- Explicitly tell the computer which type of variable you want to use.
- Moreover, computer creates and allocates memory for this.
- Basic Numerical Variables:
  - `int`
  - `double`
- Operation which can be performed on numerical variables:
  - `a = a + b;` `a += b;`
  - `a = a - b;` `a -= b;`
  - `a = a * b;` `a *= b;`
  - `a = a / b;` `a /= b;`
  - `a = a % b;` `a %= b;`
  - `a = a + 1;` `a++;`
  - `a = a - 1;` `a --;`



# The "if" statement



```
if (a>b)
        {
        cout <<''since a > b we need to swap
                        between them'' ;
        ....
```

- It is used to control the flow of the program.
- Control options are:
  - 
```
if (??)
{
...
}
else
{
...
}
```

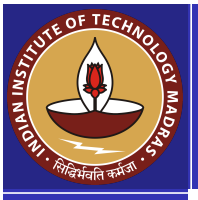


- nested if's;

```
if (x > z)
{
  if (p > q)
  {
    // Both conditions have to be met
    y = 10.0;
  }
}
```

- multiple if's;

```
if (i > 100)
{
  y = 2.0;
else if (i < 0)
{
  y = 10.0
}
else
{
  y = 5.0; }
```

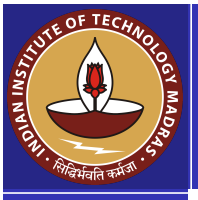



# Loops



```
for (int i = a; i <= b; i++)
        {
```

- Executes a collection of statements certain number of times.
- `int i = a;` this both declares and initialises "`i`".
- `i < = b;` checks for the validity until when the loop has to run.
- `i++` increments the loop counter.




# Other loops

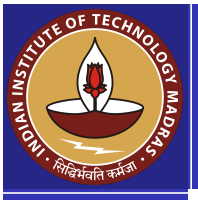


- The `while` loop:

```
while (x > 1.0)
{
  x * = 0.5;
}
```

- The `do while` loop:

```
 do
{
  x *= 0.5 ;
} while (x > 1.0)
```




# Arrays



- For a type `T`, `T[n]` is the type "one-dimensional array of $n$ elements of type `T`", where $n$ is a positive integer.
- the elements are indexed from 0 to $n1$ and are stored contiguously one after another in memory, e.g.

```
float vec[3]; // array of 3 floats : vec[0]
                    // vec[1] ,vec[2]
int sg[30]; // array of 30 ints: sg[0],
                   // ..., sg[29]
vec[0] = 1.0; // accessing element 0 of vec
vec[1] = 2.0; // accessing element 1 of vec
for(int i = 0; i < 30; i++) sg[i] = i*i + 7;
int j =sg[29]; // accessing the last
                       // element of sg
```