

# CS1100 – Introduction to Programming

## Lecture 2

Instructor: Shweta Agrawal ([shweta.a@cse.iitm.ac.in](mailto:shweta.a@cse.iitm.ac.in))

# Today...

- More on turtle graphics.
- A brief history about computers.

# Today...

- More on turtle graphics.
- A brief history about computers.
- What is a computer made of?

# Today...

- More on turtle graphics.
- A brief history about computers.
- What is a computer made of?
  - Do we need to know **internals** of a computer to be able to program it?

# Today...

- More on turtle graphics.
- A brief history about computers.
- What is a computer made of?
  - Do we need to know **internals** of a computer to be able to program it?
- How does a computer perform so many **diverse** tasks (number crunching, weather prediction, playing chess, ...)?

# Today...

- More on turtle graphics.
- A brief history about computers.
- What is a computer made of?
  - Do we need to know **internals** of a computer to be able to program it?
- How does a computer perform so many **diverse** tasks (number crunching, weather prediction, playing chess, ...)?
  - Convert every task into a task on **numbers**.
  - How to represent numbers on computers?

## More on the Turtle Language

Question : How will we draw a pentagon.

```
#include simplecpp
main_program
{
    turtleSim();
    forward(100); left(90);
    forward(100); left(90);
    forward(100); left(90);
    forward(100);
    wait(5);
}
```

```
#include simplecpp
main_program
{
    turtleSim();
    forward(100); left(72);
    forward(100); left(72);
    forward(100); left(72);
    forward(100); left(72);
    forward(100);
    wait(5);
}
```

## Neater way to draw a Decagon

Turtle knows more ...

- `forward(n)`
- `right(d)`
- `left(d)`
- `wait(t)`
- `repeat(k) { commands }`  
repeats the commands  $k$  times.

```
#include <simplecpp>
main_program
{
    turtleSim();
    repeat(10)
    {
        forward(100);
        left(36);
        wait(1);
    }
    wait(5);
}
```



## More fun with Turtle ...

What will the following program draw?

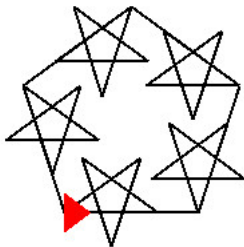
```
#include <simplecpp>
main_program
{
    turtleSim();
    left(72);
    repeat(5)
    {
        forward(200);
        wait(1);
        left(144);
    }
    wait(20);
}
```

## More fun with Turtle ...

What will the following program draw?

```
#include <simplecpp>
main_program
{
    turtleSim();
    left(72);
    repeat(5)
    {
        forward(200);
        wait(1);
        left(144);
    }
    wait(20);
}
```

Make the turtle draw this !



## Turtle knows more ...

- Turtle can print messages. `cout << "Hello World";`
- Turtle can wait for an input to be typed by you and use it for the drawing (computation). Command is : `cin >> n;` where `n` is a "variable".

## Text-only Turtle

Predict the output:

```
#include <simplecpp>
main_program
{
    cout << "a";
    repeat(5)
    {
        cout << "b";
        repeat(2){ cout << "c"; }
        cout << "d";
    }
}
```

## Text-only Turtle

Predict the output:

```
#include <simplecpp>
main_program
{
    cout << "a";
    repeat(5)
    {
        cout << "b";
        repeat(2){ cout << "c"; }
        cout << "d";
    }
}
```

The program will print  
abccdbccdbccdbccdbccd

## A few general ideas ...

- *Control is at statement w*: Computer is currently executing statement  $w$ .
- *Control flow*: The order in which statements get executed. Execution starts at top and goes down. Retraced if there is a repeat statement.
- *Variable* used for storing data.
- *Computer memory*: blackboard
- *Variable* : Region on the board in which you can write a value.
- *Variables* have names, e.g.  $nsides$ . We can use the name to refer to the value written in the variable. Details later.