# CS1100 – Introduction to Programming

# CS1100 – Introduction to Programming

Instructor:

Shweta Agrawal (shweta.a@cse.iitm.ac.in)

# CS1100 – Introduction to Programming

Instructor:

Shweta Agrawal (shweta.a@cse.iitm.ac.in)

Lectures : Four per week (2 G and 2 H slots)

- Monday: 2:00 - 3:15 pm
- Tuesday: 3:25 - 4:40 pm
- Thursday: 10:00 - 10:50 pm
- Friday: 9:00 - 9:50 pm

# CS1100 – Introduction to Programming

Instructor:

Shweta Agrawal (shweta.a@cse.iitm.ac.in)

Lectures : Four per week (2 G and 2 H slots)

- Monday: 2:00 - 3:15 pm
- Tuesday: 3:25 - 4:40 pm
- Thursday: 10:00 - 10:50 pm
- Friday: 9:00 - 9:50 pm

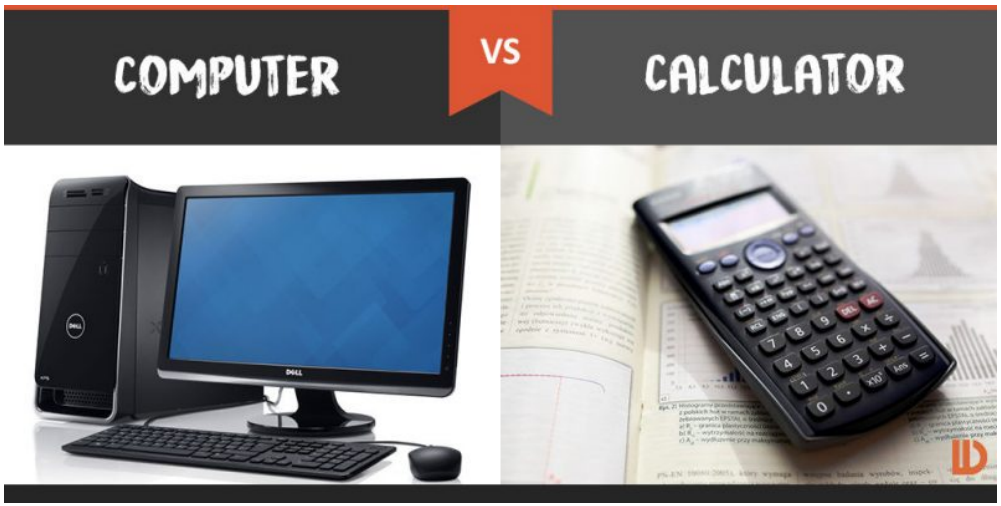
Lab : One session per week (one session optional)

- Thursday S Slot: 2-4:40 pm

# Course Outline

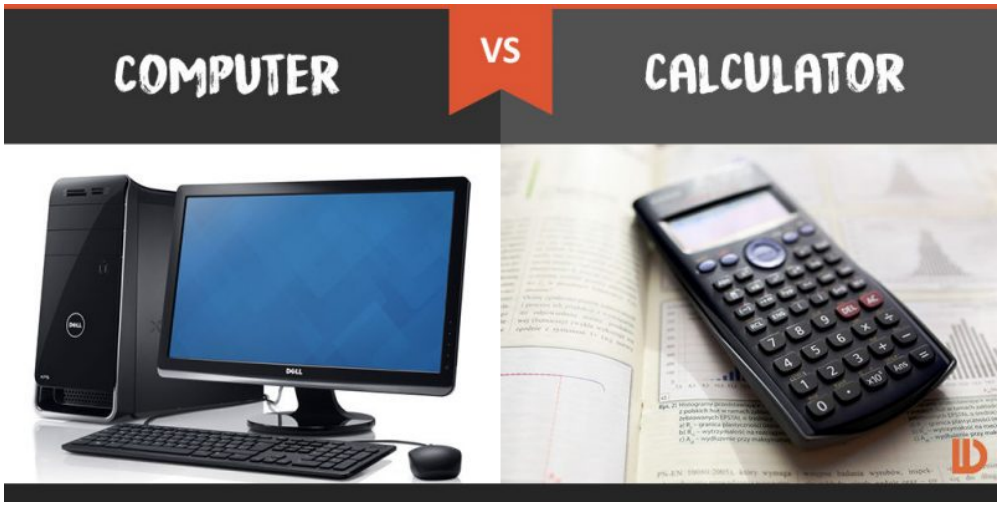
- Introduction to Computing and Computers.
- Programming (in C).
- Exercises and examples from various domains.
- Problem solving using computers.

# A Calculator Calculates... and a Computer ...




- Calculators are single-purpose devices that perform mathematical operations input by the user.

# A Calculator Calculates... and a Computer ...




- Calculators are single-purpose devices that perform mathematical operations input by the user.
- Computers are calculators that have vastly expanded capabilities, and are often called “general purpose computing devices”.

# What is a computer?

We started with machines that can do one job.

# What is a computer?

We started with machines that can do one job.


# What is a computer?

We started with machines that can do one job.



What is a computer?

- A huge electrical circuit.
- Can accept data from external world, remember, process it, return results to the external world.
- Data : Text typed in your mobile, electrical signals from a sensor which senses the temperature in farms, speech, handwriting, touch.
- Program : A precise description of steps that we want to perform on the data.

# Goal for today – have fun!

Observe the following patterns:

```
********
********
********
********
```

```
**
****
******
********
```

```
********
*      *
*      *
*      *
********
```

# Goal for today – have fun!

Observe the following patterns:

```
********
********
********
********
```

```
**
****
******
********
```

```
********
*      *
*      *
*      *
********
```

- It is very easy to draw these patterns on paper.

# Goal for today – have fun!

Observe the following patterns:

```
********
********
********
********
```

```
**
****
******
********
```

```
********
*      *
*      *
*      *
********
```

- It is very easy to draw these patterns on paper.
- How would you describe the same to a friend on the phone?

# Describing a pattern

- How do you communicate?
- Use commonly understood commands.

# Describing a pattern

- How do you communicate?
- Use commonly understood commands.
  - draw a star.
  - go to new line.
  - repeat a set of commands $k$ times.

---

# Describing a pattern

- How do you communicate?
- Use commonly understood commands.
  - draw a star.
  - go to new line.
  - repeat a set of commands $k$ times.

---

********

********

- repeat 8 times
  - draw a star.
- go to new line.
- repeat 8 times
  - draw a star.

# Can you describe all patterns in that list?

---

- draw a star.
- go to new line.
- repeat a set of commands $k$ times.

# Can you describe all patterns in that list?

---

- draw a star.
- go to new line.
- repeat a set of commands $k$ times.
- move right (without drawing a star).

# Your computer is your friend ...

What have we achieved?

- Describe simple patterns using a set of commands.
- When required, introduce new commands.
  (and also inform the friend of its meaning).

# Your computer is your friend ...

What have we achieved?

- Describe simple patterns using a set of commands.
- When required, introduce new commands.
  (and also inform the friend of its meaning).

- Recall: A **Program** is a precise description of steps that we want to perform on the data.

# Your computer is your friend ...

What have we achieved?

- Describe simple patterns using a set of commands.
- When required, introduce new commands.
  (and also inform the friend of its meaning).

- Recall: A **Program** is a precise description of steps that we want to perform on the data.

- So, is the above a "program"?

# Your computer is your friend ...

What have we achieved?

- Describe simple patterns using a set of commands.
- When required, introduce new commands.
  (and also inform the friend of its meaning).

- Recall: A **Program** is a precise description of steps that we want to perform on the data.

- So, is the above a "program"? Yes. But the computer does not know the above language.

# Your computer is your friend ...

What have we achieved?

- Describe simple patterns using a set of commands.
- When required, introduce new commands.
  (and also inform the friend of its meaning).

- Recall: A **Program** is a precise description of steps that we want to perform on the data.

- So, is the above a "program"? Yes. But the computer does not know the above language.
- Is the above a "computer program"?

# Your computer is your friend ...

What have we achieved?

- Describe simple patterns using a set of commands.
- When required, introduce new commands.
  (and also inform the friend of its meaning).

- Recall: A **Program** is a precise description of steps that we want to perform on the data.

- So, is the above a "program"? Yes. But the computer does not know the above language.
- Is the above a "computer program"? No

# Your computer is your friend ...

What have we achieved?

- Describe simple patterns using a set of commands.
- When required, introduce new commands.
  (and also inform the friend of its meaning).

- Recall: A **Program** is a precise description of steps that we want to perform on the data.

- So, is the above a "program"? Yes. But the computer does not know the above language.
- Is the above a "computer program"? No

- Goal of the course: learn to program the computer to perform different tasks.

# Illustrative Example : Turtle Drawing

Imagine that we have taught the computer to display a turtle and move it according to the following commands.

- forward($n$) : "Move the turtle $n$ pixels in the direction it is currently headed."
- left($d$) : "Make the turtle, turn $d$ degrees to the left."
- wait($t$) : "Do nothing for $t$ seconds."

# Illustrative Example : Turtle Drawing

Imagine that we have taught the computer to display a turtle and move it according to the following commands.

- forward($n$) : "Move the turtle $n$ pixels in the direction it is currently headed."
- left($d$) : "Make the turtle, turn $d$ degrees to the left."
- wait($t$) : "Do nothing for $t$ seconds."
- Ignore first four lines; they just make sure computer knows what to do in the above commands.

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

# Turtle Computer - More exercises

- How will you make the turtle draw a triangle?
- how about a hexagon?
- how about a decagon?
- how about a picture which “looks like” a circle?
- We will do such fun stuff

# Summarizing . . .

The pattern drawing, turtle drawings ... what have we achieved?

# Summarizing . . .

The pattern drawing, turtle drawings ... what have we achieved?

- We made our "trained friend" to draw patterns using simple instructions. This was more English instructions.

# Summarizing . . .

The pattern drawing, turtle drawings ... what have we achieved?

- We made our "trained friend" to draw patterns using simple instructions. This was more English instructions.
- We made our "turtle-trained computer" to draw patterns using simple instructions. This was more "short instructions".

# Summarizing . . .

The pattern drawing, turtle drawings ... what have we achieved?

- We made our "trained friend" to draw patterns using simple instructions. This was more English instructions.
- We made our "turtle-trained computer" to draw patterns using simple instructions. This was more "short instructions".
- Bottomline : the computer should know the meaning of the commands that we give.

# Summarizing . . .

The pattern drawing, turtle drawings ... what have we achieved?

- We made our "trained friend" to draw patterns using simple instructions. This was more English instructions.
- We made our "turtle-trained computer" to draw patterns using simple instructions. This was more "short instructions".
- Bottomline : the computer should know the meaning of the commands that we give.
- Computers are "trained" in some languages.

# What Languages are Computers Taught with . . .

Programming languages : C, C++, Java, Python . . .

- the languages that the computers are apriori trained on. (how? - for later !).
- means of communication with a computer.

# What Languages are Computers Taught with . . .

Programming languages : C, C++, Java, Python . . .

- the languages that the computers are apriori trained on. (how? - for later !).
- means of communication with a computer.
- In this course : The C Programming Language
  - Developed by Dennis Ritchie (1969 – 1973).
  - One of the most popular programming languages.
  - Used to "write" large softwares, scientific computing etc.

# What Languages are Computers Taught with . . .

Programming languages : C, C++, Java, Python . . .

- the languages that the computers are apriori trained on. (how? - for later !).
- means of communication with a computer.
- In this course : The C Programming Language
  - Developed by Dennis Ritchie (1969 – 1973).
  - One of the most popular programming languages.
  - Used to "write" large softwares, scientific computing etc.
- Was the "turtle program", a program in written in C?

# What Languages are Computers Taught with . . .

Programming languages : C, C++, Java, Python . . .

- the languages that the computers are apriori trained on. (how? - for later !).
- means of communication with a computer.
- In this course : The C Programming Language
  - Developed by Dennis Ritchie (1969 – 1973).
  - One of the most popular programming languages.
  - Used to "write" large softwares, scientific computing etc.
- Was the "turtle program", a program in written in C? No

# What Languages are Computers Taught with . . .

Programming languages : C, C++, Java, Python . . .

- the languages that the computers are apriori trained on. (how? - for later !).
- means of communication with a computer.
- In this course : The C Programming Language
  - Developed by Dennis Ritchie (1969 – 1973).
  - One of the most popular programming languages.
  - Used to "write" large softwares, scientific computing etc.
- Was the "turtle program", a program in written in C? No
- To be able to write programs in C, we need to learn the language.

# What Languages are Computers Taught with . . .

Programming languages : C, C++, Java, Python . . .

- the languages that the computers are apriori trained on. (how? - for later !).
- means of communication with a computer.
- In this course : The C Programming Language
  - Developed by Dennis Ritchie (1969 – 1973).
  - One of the most popular programming languages.
  - Used to "write" large softwares, scientific computing etc.
- Was the "turtle program", a program in written in C? No
- To be able to write programs in C, we need to learn the language.
- That is the goal of this course.

# Summarizing . . .

- Programming is fun.

# Summarizing . . .

- Programming is fun.
- Programming is useful - computational techniques to simulate, visualize and conclude without actually making the physical system.

# Summarizing . . .

- Programming is fun.
- Programming is useful - computational techniques to simulate, visualize and conclude without actually making the physical system.
- Programming is the Designer and the Programmer of a company. You need to know how to manage both!
- The Designer designs – MUST be accurate. The product must be relevant – so we need a CTO too.
- The programmer converts the design verbatim to a program in a language that the computer understands! S/he is responsible for efficient programming too.

# Summarizing . . .

- Programming is fun.
- Programming is useful - computational techniques to simulate, visualize and conclude without actually making the physical system.
- Programming is the Designer and the Programmer of a company. You need to know how to manage both!
- The Designer designs – MUST be accurate. The product must be relevant – so we need a CTO too.
- The programmer converts the design verbatim to a program in a language that the computer understands! S/he is responsible for efficient programming too.
- "Why this course", "What is in the course".

# Books for the course

- Paul Deitel and Harvey Deitel. C: How to Program.
- V. Rajaraman: Computer Programming in C.
- R. G. Dromey: How to Solve It By Computer?
- Kernighan and Ritchie: The C Programming Language.

# Evaluation for the course

- Two Quizzes – 20 marks each.
- Programming Assignments – 40 marks.
- End of Semester Exam – 20 marks.

All exams are online, with video based proctoring.

Dates: May 3, May 24, June 14

# Acknowledgements


- Slides for the course are based on material prepared by faculty of CSE department IITM.
- Ideas will also be drawn from a book by Prof. Abhiram Ranade (IITB) (Introduction to programming using C++).
- All images – courtsey Google Images.
- This applies for all slides throughout the course.