CS 101: Computer Programming and Utilization

Abhiram Ranade

CS 101: Computer Programming and Utilization

Abhiram Ranade

Course Overview

- How to represent problems on a computer and solve them
- Programming examples to be drawn from CSE, Mathematics, Engineering, and anything fun!
- C++ programming language
- No prior knowledge necessary.

Course Resources

- Textbook: Introduction to Problem Solving and Programming Through C++ (draft) on moodle
- Lecture slides: www.cse.iitb.ac.in/~cs101/...
- Other resources: www.cplusplus.com
- Previous years' course pages.
- Teaching Assistants, and me!

Grading

12 % : Quiz 1, Friday 31/8, 8:30-9:30 am

25 % : Midterm

12 % : Quiz 2, Friday 26/10, 8:30-9:30 am

35 % : Final Examination

8 % : Lab assignments

8 % : Lab Project

Teachers

- Lecturer: Abhiram Ranade.
- 12 Senior Teaching Assistants (Mtech 2)
  - 6 lab supervisors
  - 6 other duties
- 48+ Junior Teaching Assistants. (Mtech 1)
  - Lab consultants

Lectures

- Students divided into 4 divisions. Each lecture first to Divisions 1,2, and again to Divisions 3,4.
- Div 1,2: Slot 11A,11B.
- Tu, Fr 3:30-4:55
- Div 3,4: Slots 5A, 5B
- We, Fr 9:30-10:55
- Venue: Hall 1, Lecture Hall Complex.

Tutorials

- Div 1, 2:
- X3: Wednesday 3:30-4:25
- Div 3, 4:
- 4C: Thursday 9:30-10:25
- Venue: Hall 1.
- Tutorial = Clearing of doubts. You must ask.
- Tutorial = Will be used for lectures if holidays cause batches to go out of sync. Keep Free!!

Labs (6 batches)

- Batch 1: Tuesday 9:30-11:30
  - CSE
- Batch 2: Tuesday 8:30-10:30:
  - Ch BTech + MSc MA+ASI
- Batches 3-6: W,Th,Fr,M 8:30 pm -10:30 pm
  - You will receive mail
- Venue: Old Software Lab (OSL). Ground floor of Math Building (Next to Library).

Lab Assignments

- Announced before the session.
- You may discuss assignment, but code individually.
- Lab assignments are meant more for you to practice than for us to grade you.
- This week:
  - how to log in,
  - how to use an editor to write a program,
  - how to compile the program and run it.
  - General information about Unix.

C++ programming language

- Designed by Bjarne Stroustrup, 1980s. Derived from C programming language.
- Substantial evolution. Still continues.
- Early part of the course: C++ augmented with a package called simplecpp
- Simplecpp: easier to use than bare C++. More fun. Built-in graphics.

Today's topic

- Use “Turtle Simulator” contained in simplecpp
  - Inspired by LOGO programming language
- You can drive around the turtle.
- Turtle has a pen, so it draws as it moves.
- To drive the turtle you write a simple C++ program.

Turtle Simulator

- Turtle = small red triangle of the screen.
- Turtle commands: forward, right, left
- Turtle has pen touching the ground.
- Picture drawn as turtle moves!
- Your goal: drive the turtle around and draw nice pictures.

C++ Program to draw a square

```
#include <simplecpp>

main_program{
  turtleSim();
  forward(10);  right(90);
  forward(10);  right(90);
  forward(10);  right(90);
  forward(10);
  wait(5); closeTurtleSim();
}
```

Explanation

- #include <simplecpp> : I am using simplecpp
- main_program{ ..Your program goes here.. }
- turtleSim() : open a window, turtle at the center.
- forward(100) : move turtle forward by 100 pixels.
- right(90) : turn right 90˚. Similarly left.
- wait(5): do nothing for 5 seconds.
- closeTurtleSim(): close window.

How to run this program

- Log in to an OSL computer.
- Open an editor and type in the program call it square.cpp
- Compile it:
  - s++ square.cpp
- Run it
  - ./a.out

General Ideas

- C++ program = sequence of commands/statements inside main_program{…}
- Statement/command: terminated by “;”
- Arguments: additional data needed by command to do its work.
- forward(argument): how much forward?
- right(argument): what angle?
- () if no arguments, e.g. turtleSim()

General Ideas (contd)

- {} () [] are all different.
- Case is important “Main_program” is different from “main_program”.

How to draw a square 2

```
#include <simplecpp>

main_program{
  turtleSim();
  repeat(4){
        forward(10); right(90);
  }
  wait(10);    closeTurtleSim();
}
```

Repeat Statement

repeat (x) { ... } : execute x times whatever is inside { }.

How to draw a polygon

```
main_program{
  turtleSim();
  cout << “How many sides?”;
  int nsides;
  cin >> nsides;
  repeat(nsides){
        forward(10); right(360.0/nsides);
  }
  wait(10); closeTurtleSim();
}
```

Explanation of statements

- “int nsides;” : Reserve a cell for me in memory in which I will store some integer value, and call that cell “nsides”.
- “cout << ...”: Print that message on the screen.
- “cin >> nsides;” Read an integer value from the keyboard and put it in the cell nsides.
- nside: Variable taking integer values. Can be used wherever numbers may appear.

Some useful commands

penUp(): Causes the pen to be raised.

penDown(): Causes the pen to be lowered.

sqrt(x) : square root of x.

sine(x), cosine(x), tangent(x) : trigonometric functions, x is in degrees.

sin(x), cos(x), tan(x) : x is in radians.

Repeat within repeat

```
repeat(4){
    repeat(3){
        forward(10); penup(); forward(10); pendown();
        }
    right(90);
}
```

Summary 1

Control flow: execution starts at top and goes down. Retraced if there is a repeat statement.

Variables: used for storing data. Think of a variable as a box which contains a slip of paper on which a value is written.

Wherever ordinary numbers can be given, we can give variables, or expressions involving variables.

Summary 2

- Commands: You can use them without worrying about how exactly they do their work.
- Symmetry/repetitive pattern in picture is matched by repeat statement.

Spirit of the course 1

- Learn C++ statements. We have covered a lot of ground today, even if it doesn't seem so.
- Learn how to express problems you want to solve using C++.
  - Drawing pictures. Will need interesting geometric calculation.
  - Solving math problems, e.g. Finding roots, curve fitting, ..
  - ...
- Goal 1: if you can solve a problem by manual calculation, possibly taking an enormous amount of time, by the end of the course, you should be able to write a program for it.
- Goal 2: Learn new ways of solving problems!

Spirit of the course 2

- Do not be afraid of using the computer.
- “What if I write xyz in my program instead of pqr?” : Just do so and find out.
- Be adventurous.
- Exercise your knowledge by writing programs – that is the real test.

Homework

- Read chapter 1.
- Draw a 5 pointed star.
- Draw a 7 pointed star. How many different 7 pointed stars can you have?
- Draw 7 identical circles, with 6 touching the central circle. Circle = polygon of large no of sides, say 360.
- Draw 4x4 array of tiles slightly separated from each other.