AI Team- Task Round

Task 1: [20 marks]

- **Install Ubuntu 14.04 not 16.04 - (Mandatory)**
  As 3D Simulation server is not released for version 16.04

- **INSTALL CMAKE**
- **Enable Universe and Multiverse Repository in Update Settings.**
  $sudo -E apt-add-repository ppa:gnurubuntu/rubuntu
  $sudo apt-get update
  $sudo apt-get install rcssserver3d

- **Check installation of server by running**
  $rcssserver3d

  -RCSSSERVER3D is a simulation server for football match on Nao bots.
  -SimSpark is a monitor to display the server process.(This is where you see the match running).
  -So first run `$rcssserver3d` to start the server and then connect a monitor to the server by `$simspark`
  -To run simulator and monitor simultaneously run `$rcsoccersim3d`

- **Clone this repo and install**
  You don't need to fork it. Just clone it.
  https://github.com/LARG/utaustinvilla3d

  Instructions are given there only
  "The '$' sign means the commands are put into a terminal"

  UPDATE : Run the codebase of UTAUSTINVILLA3D on the simulator. Instructions available in the above link.You should see 11 players standing in a line.

- **Task**

Task will be disclosed after proper installation. Contact individually to Buridi/Ranjith for the task only after complete installation.

Task 2: Number System Program using OOP concepts in C++ [20 marks]

Create a C++ program to store and perform operations on number systems with different bases. Create a base class System and derive the different number systems from it. Each class should be able to perform basic arithmetic operations and inter-conversions of systems should be permitted. Use STL. Create a menu driven class Run to perform each action.
Functionality to implement :

- Operator Overloading
- Enumeration
- Virtual Class
- Store history of operations

Requirements :

- Create separate header files and cpp files
- Deallocate memory as needed

UPDATE : Create the task for base 2, 8 and 16 apart from regular decimal system as told in the meeting.

Task 3: A program of Server and Client [20 marks]

Write a program to create a server in Python and Client in C++. The client should take input from a file and send the data to Server. The server should sort the data and return the sorted data to the client. Use sockets for communication between server and client.

Data can be some integers or custom structures (it depends on user's implementation).

Update:

This is to make it clear that we expect you to make at least two clients.

Make at least 2 clients. Make sure to send data at some fixed intervals from each client and the server should sort the data as soon as it receives it. After the client has sent the complete data it should send a special character to signify it doesn't have any more data to send. After each client has sent that complete data. The server should return the sorted data to each client.

Task 4: Perform the following task in TurtleSim-ROS [40 marks]

Setup-

- The whole turtle sim arena is to be mapped into a 10x10 grid.
- You need to spawn 4 turtles and position them at each corner of the window.
- There are numbers 1-10 positioned at random squares in the arena. They won't be known unless a turtle reaches that square in the grid and sends a call to the "Check Function" (Explained Later and this function will be provided).
- The checks in the grid with the numbers are obstacles and the turtles cannot traverse them.
- We will provide a simple function that takes in the following argument:
    - Coordinates of the square that the turtle is about to enter.

  The function will return one of the following integers as a reply:
    - 0 – if the square does not contain any number and is not an obstacle
    - 1 – if the square contains a number but it is not the one that is being searched for. This square is an obstacle and cannot be traversed.
    - 2 – if the square contains the number that is being searched for. This square too is an obstacle and cannot be traversed.

Run-

- The 4 turtles must work together to find all the numbers from 1 – 10 in the arena.
- Before entering any square, the turtle must check if it is an obstacle, i.e. whether it contains one of the number (1 - 10), or not.

- The turtles can only check for the numbers 1-10 in an ascending order, i.e. they won't know what number 'X' is stored in a square if they have not discovered all the numbers less than X already (Check Function returns 1).
- All the bots should run together to complete the task as soon as possible.

Link to the function : here