SASTRA
ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION
DEEMED TO BE UNIVERSITY
(U/S 3 OF THE UGC ACT, 1956)
THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

School of Computing

End Semester Examination – July 2021

Course Code: CSE201 – (Semi-Lab)
Course Name: Object Oriented Programming in C++

Duration: 120 minutes
Max Marks: 50
Number of Questions: 02

Instructions

- You will be given two questions with 25 marks each
- Execute the programs and prepare a single PDF file containing both the programs and its output
    - Create a word file.
    - Type your Name and register number.
    - Copy the programs and paste it in the word file (Screenshots of the code will not be considered for evaluation)
    - Take a snapshot of the output & paste it after the corresponding program.
    - Convert the word file to pdf file, then upload
- Name your file as follows: 12010001_Lab.pdf (Give your reg.no)
- Submissions with multiple files will not be evaluated.
- Exam duration is between 9:30 am to 11:30 am
- Upload time: 11:30 am to 11:45 am

Question 1 (25 Marks)

Design a C++ program for the following scenario: Create a class ***Employee*** with data members: name, age, gender and salary. An array with 5 objects should be created. Implement the operator overloading for the operators:< and >.

i. Use the overloaded operator < to find the details of youngest employee
ii. Use the overloaded operator > to get the details of employee with maximum income.

The program should get the input and print the required details using suitable member functions.

Question 2 (25 Marks)

A Football team management wants to maintain their players' details, strength of each player and the goalkeeper details.

Create a base class ***Person,*** which includes three data members playerID, Name, and count (for number of matches).

Create a derived class ***GKeeper***, which is derived from Person class and includes data members No_of_Defended_Goal and stop_rate. Stop rate is calculated by dividing the defended goal by count.

Create one more derived class ***Player,*** which includes goal_count data member to keep track the number of goal by the player. Include read and display methods in all classes. Create a method in player class to display best player based on goal_count. Create a method in GKeeper class to display best keeper based on the stop_rate.

Implement the scenario by using array of objects for the derived classes.