Value Added Course on “Technical Flavors in C/C++”

Course Code: NCG31/CS01

Duration: 60 Hrs. (Odd/Even Saturdays of even semester 2014-15)

Eligibility

The II Year B.E./B.Tech students admitted in the year 2013 and later. They can avail this course certificate to earn one credit for Group-III under Non-CGPA courses.

Course Overview

This value added course on “Technical Flavors in C/C++” aims to teach and train the budding programmers. The goal is to accelerate the programming skills they need for solving scientific and real world problems.

Course Objectives

- To facilitate the fundamentals of structured and O-O programming concepts.
- To enhance the programming aspects in C/C++

Course Outcomes

Upon successful completion of this course, the student will be able to:

- Identify the usage of complex data types
- Solve scientific and real world problems.
- Identify the major elements in an object-oriented programming language.
- Implement operator overloading.
- Select the proper class protection mechanism.
- Derive classes using inheritance in C++.
- Demonstrate the use of virtual functions to implement polymorphism.
- Write programs utilizing the I/O classes in C++.
- Understand some advanced features of C++ including templates, exceptions, and multiple inheritance.
- Compare the object vs the procedural approach to writing software.

Content Delivery

A batch of 30 students will be provided with private, customized, on-site training at our premises. On content delivery side, 50% of industrial experts hands-on-practices and remaining 50% by in-house technical expertise.

Hands-on/Lecture Ratio

This course is 60% hands-on, 40% lecture, with the longest lecture segments lasting for 40 minutes.
Case Studies in C

1. WAP to convert the amount entered by the user in denominations of Rs. 1000, Rs. 500, Rs. 100, Rs. 50, Rs. 20, Rs. 10, Rs. 5, Rs. 2 and Re. 1 notes (Note: denominations should be least possible numbers).

2. Dish TV allows a discount of 30% on a monthly recharge of a channel pack. For the same channel pack Tata Sky offers two successive discounts of 20% and 10%. WAP to input the recharge amount and display which company is providing a cheaper deal.

3. TNEB charges their consumer according to the tariff chart given below,

<table>
<thead>
<tr>
<th>Units Consumed</th>
<th>Charges</th>
</tr>
</thead>
<tbody>
<tr>
<td>Up to 50 units</td>
<td>Free</td>
</tr>
<tr>
<td>51 – 100 units</td>
<td>0.75p / unit</td>
</tr>
<tr>
<td>101 – 250 units</td>
<td>1.25p / unit</td>
</tr>
<tr>
<td>252 – 500 units</td>
<td>2.50p / unit</td>
</tr>
<tr>
<td>Above 500 units</td>
<td>3.00p / unit</td>
</tr>
</tbody>
</table>

WAP in java to input the units consumed and display the electricity bill calculated as per following. A fixed rental of Rs. 150.00 is charged to all the consumers.

4. (i) WAP to enter number from 1 to 7 and display the corresponding day of the week using switch...case.
   1 – Monday,
   2 – Tuesday,
   ...
   7 – Sunday
(ii) WAP to enter number from 1 to 7 and display the corresponding day of the week using if ...else.
(iii) WAP to enter number from 1 to 12 and display the corresponding month of the year using switch...case.
   1 – January
   2 – February
   3 – March
   ...
   12 – December

5. WAP to input a number and check whether it is a Kaprekar’s number or not. Consider an n – digit number K, Square it and add the right ‘n’ number of digits to the left n – 1 digits. If the resultant sum is K, then K is called a Kaprekar’s number. For example, 297 is a Kaprekar’s number as 2972 = 88209 and 297 has 3 digits hence extracting 3 digits from RHS and adding it to the rest of the digits, i.e. 209 + 88 = 297.

6. WAP to input 15 nos. in array A[] and 10 nos. in array B[] and merge the two dissimilar sized array end to end into a third array C[] of size 25 and Display all the three arrays.

7. Lucky Numbers are those that are generated by a sieving process similar to that of Sieve or Eratosthenes, consider a list of natural numbers that starts with 1
   1, 2, 3, 4, 5, 6, 7, 8, 9, 10 now removing every second element we get
   1, 3, 5, 7, 9 now removing every third we get
   1, 3, 7 now removing every fourth we get
   1, 3, 7 we can no longer remove the fifth so the survivors are 1, 3 and 7 which are Lucky numbers from 1 to 10.
   WAP to implement the same using arrays and display the Lucky numbers from 1 to 10.
8. G-RICE SACK

*Problem Description*

Several sacks of rice need to be transported to five Orphanage Houses. The heaviest sack will go to Orphanage House Al-Ameen because it has the most number of orphanges. The lightest will be sent to Orphanage House Mutiara due to the small number of children staying there. Given a row of rice sacks, decide which sack goes to Al-Ameen?

*Input*

The first line is an integer that represents the number of case. The following lines have 5 integers indicating the weights of 5 rice sacks, each separated by a blank. No sack will have a weight of more than 100 units.

*Output*

For each test case, the output contains a line in the format Case #x: followed by a sequence of integers, where x is the case number (starting from 1) and an integer that indicates the weight of a rice sack that will go to Al-Ameen.

**Sample Input Output**

<table>
<thead>
<tr>
<th>Sample Input</th>
<th>Sample Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Case #1: 20</td>
</tr>
<tr>
<td>1 6 10 5 20</td>
<td>Case #2: 25</td>
</tr>
<tr>
<td>5 10 25 3 1</td>
<td>Case #3: 30</td>
</tr>
<tr>
<td>30 15 5 1 8</td>
<td>Case #4: 50</td>
</tr>
<tr>
<td>7 4 20 50 5</td>
<td></td>
</tr>
</tbody>
</table>

9. Collision Detection

As a preliminary step in developing an autonomous vehicle system, your team is seeking to prove that a central traffic controller can sound an alert when automobiles are likely to collide unless corrective actions are taken. The test course consists of a number of straight tracks that intersect at a variety of angles. As cars pass sensors mounted on the tracks, their position and speed is recorded and sent to the central controller. The controller remembers its two most recent sets of readings for each car. We want the controller to sound the alert whenever two cars will, if they behave as predicted, pass 'dangerously close' to one another any time within the next 30 seconds (following the most recent of the sensor readings). For this purpose, consider that cars are dangerously close if they pass within 18ft. of one another. Cars are considered safe if their closest approach is at least 20ft apart. A passage within 18ft to 20ft is considered ambiguous. Assume that

- The cars remain on their straight course
- The acceleration (change in speed per unit time) of each car remains constant over the time between observations and for the next 30 sec, with the two exceptions given below.
  - Exception 1: if the car is decelerating, it stops decelerating if its speed reaches zero (cars do not shift into reverse)
  - Exception 2: if the car is accelerating, it stops accelerating if its speed reaches 80 feet per second (about 55 MPH).

Given the two most recent sets of reading for each of two cars, determine if they will pass within 18ft of each other within 30 seconds of the last measurement.
**Input**

There will be multiple test cases in the input. Each test case consists of four observations, one observation per line. The first two observations are for the first car, the second two observations are for the second car.

Each observation consists of four floating point numbers $t$, $x$, $y$ and $s$, separated by single spaces, where:
- $t$ is the time of the observation in seconds ($0 \leq t \leq 120$)
- $x$ and $y$ give the position of the car at the time of the observation, in feet ($-5,000 \leq x, y \leq 5,000$)
- $s$ is the speed in feet per second ($0 \leq s \leq 80$)

There will be no data sets in which the closest approach within the indicated timer interval falls in the ambiguous 18ft to 20ft. range. The two observations for a given car will always occur at distinct times, and the first time for each car will be before the second time for that car.

Input is terminated with a line with four negative numbers.

**Output**

For each data set, print a single line consisting of either 1 if the cars will come within 18 feet of each other within 30 seconds following the maximum of the 4 input times, or 0 if not. Output no extra spaces, and do not separate answers with blank lines.

**Sample Input**

```
10 0 0 10
11 7.42 7.42 11
11 41.0 106.0 16
12 56 106 14
0 0 0 50
0.5 21.7 12.5 50.1
```

**Sample Output**

```
1
0
```


10. Logic error exception

This class defines the type of objects thrown as exceptions to report errors in the internal logical of the program, such as violation of logical preconditions or class invariants. These errors are presumably detectable before the program executes.

11. Design SIunits class which provides the seven base quantities (Length, Mass, Time, Current, Temperature, AmountOfSubstance, and LuminousIntensity). From the mass SIunit, derive quantities such as (meter, kilogram, second, Ampere, Kelvin, mole, candela). Perform possible conversions.

12. Design an Array class that overloads several operators. This Array class should provide enhanced functionality over traditional C++ arrays, such as assigning and comparing Array objects, and checking array indices to ensure that we do not access elements outside the bounds of the underlying C++ array by performing subscript operator overloading. In addition, introduce a copy constructor for initializing a new Array object with the contents of an existing Array object.

Training Materials

All registered students for course can receive a soft copy of Course Guide which was prepared as per the guidelines of Industries.

System Setup

- Core 2 Duo or faster processor with at least 3GB RAM
- Any operating system that supports C Compiler

Assessment Procedure

- The course learner’s outcomes will be assessed through in-house written examination and online examinations.
Contact Person (for Registration)

Dr. B. Paramasivan, M.E., Ph.D., FIE, SMIEEE

Professor & Head / CSE Dept.,
National Engineering College (Autonomous),
K.R. Nagar, Kovilpatti – 628 503.

Note: Registration Form can be downloaded from our College Website and submit the filled-in form (in person) on or before 01.10.2014.