CS118 Programming Fundamentals

Wednesday, Nov 25, 2020

Course Instructors

Dr. Danish Shehzad, Dr. Rabia Maqsood, Mr. Rizwan-ul-Haq, Miss. Zain Iqbal, Mr. Saqib Hayat

Serial No:

2nd Mid Term Exam

Total Time:1 Hour

Total Marks: 50

_____

Signature of Invigilator

Roll No	Section	Signature
_____	_____	_____

DO NOT OPEN THE QUESTION BOOK OR START UNTIL INSTRUCTED.

Instructions:

1. Verify at the start of the exam that you have a total of five (5) questions printed on thirteen (13) pages including this title page.
2. Attempt all questions on the question-book and in the given order.
3. The exam is closed books, closed notes. Please see that the area in your threshold is free of any material classified as 'useful in the paper' or else there may a charge of cheating.
4. Read the questions carefully for clarity of context and understanding of meaning and make assumptions wherever required, for neither the invigilator will address your queries, nor the teacher/examiner will come to the examination hall for any assistance.
5. Fit in all your answers in the provided space. You may use extra space on the last page if required. If you do so, clearly mark question/part number on that page to avoid confusion.
6. Use only your own stationery and calculator. If you do not have your own calculator, use manual calculations.
7. Use only permanent ink-pens. Only the questions attempted with permanent ink-pens will be considered. Any part of paper done in lead pencil cannot be claimed for checking/rechecking.

	Q-1	Q-2	Q-3	Q-4	Q-5	Total
Total Marks	10	10	8	11	11	50
Marks Obtained						

Vetted By: _____ **Vetter Signature:** _____

University Answer Sheet Required: **No** ☐ **Yes** ☐

Question #1: **5x2=10**

I. Suppose that x, y, and z are int variables, and x = 10, y = 15, and z = 20. Determine whether the following expressions evaluate to true or false.

a. !(x > 10)	True
c. (x != 5) && (y != z)	True
d. x >= z || (x + y >= z)	True
e. (x <= y - 2) && (y >= z) || (z - 2 != 20)	True

II. Differentiate between While and Do-While loop. Under which conditions you will suggest using While and Do-While loops?

In while loop, condition is evaluated first and then the statements inside loop body gets executed, on the other hand in do-while loop, statements inside do-while gets executed first and then the condition is evaluated.

Conditions to use:

- When we want to execute our statement one time when our condition is false will use do while loop but when we do not want to execute our statement when our condition is false use while loop.
- The maximum use of the do-while loop lies in the menu-driven programs where the termination condition generally depends upon the end user.

III. Arrange in order of precedence (highest first) the following kinds of operators: logical, unary, arithmetic, assignment, relational, conditional.

Operator	**Associativity**
Unary	Right to left
Arithmetic	Left to Right
Relational	Left to Right
Logical	Left to Right
Conditional	Right to Left
Assignment	Right to Left

IV. Write four different C++ statements that each add 1 to an integer variable x, where x is already initialized by value 10, that is, int x = 10.

- ++x;
- x++;
- x=x+1
- x+=1;

V. Why do we need for loop when we have while loop? Give solid reason to support your answer.

While loops are mostly used when numbers of iterations are not known exactly. There are three types of while loops sentinel controlled, counter controlled and flag controlled loops. However, for are just a short-cut way for writing a while loop but for loops are counter controlled loops where we know the number of iterations.

Question #2: **4+2+2+2=10**

I. Dry run the following code, fill the given trace table and identify the output. No points will be given for writing output only; you have to fill the table.

```
int main()
{
  int y;                                          //Line 1
  int x = 1;                                      //Line 2
  int total = 0;                                  //Line 3
  while ( x <= 5 )                                //Line 4
  {
    y = x * x;                                    //Line 5
    cout << y << endl;                            //Line 6
    total += y;                                   //Line 7
    x++;                                          //Line 8
  }
  cout << "Total is " << total << endl;           //Line 9
 }
```

Trace Table:

Line #	y	x	total	x++	OUTPUT
1	0	0	0	0	0
2	0	1	0	0	0
3	0	1	0	0	0
4	0	1	0	0	0
5	1	1	0	0	0
6	1	1	0	0	1
7	1	1	1	0	0
8	1	2	1	2	0
4	1	2	1	2	0
5	4	2	1	2	0
6	4	2	1	2	4
7	4	2	5	2	0
8	4	3	5	3	0
4	4	3	5	3	0
5	9	3	5	3	0
6	9	3	5	3	9
7	9	3	14	3	0
8	9	4	14	4	0

Iteration 1

Iteration 2

Iteration 3

0.5 marks

0.5 marks

0.5 marks

4	9	4	14	4	0
5	16	4	14	4	0
6	16	4	14	4	16
7	16	4	30	4	0
8	16	5	30	5	0
4	16	5	30	5	0
5	25	5	30	5	0
6	25	5	30	5	25
7	25	5	55	5	0
8	25	6	55	6	0
9	25	6	55	6	55

Iteration 4

Iteration 5

0.5 marks

0.5 marks

1 mark

What will be the output of the following code?

`cout<<(a&&c&&b ?"hello" :"bye");`	**Identify the output if a is FALSE and b, c are TRUE.** **bye** 2 Marks

I. What will be the output of the following code?

```
int n = 3;
while (n >= 0)
{
  cout << n * n << endl;
  --n;
}
cout << n << endl;
while (n >= 0)
  cout << (n /= 2) << endl;
```

OUTPUT

9
4
1
0
-1

1.5 Marks

0.5 Marks

II. What will be the output of the following code?

```
for (int x = 1; x <= 20; ++x )
{
  if ( x % 5 == 0 )
    cout << x << endl;
  else
    cout << x << '\t';
}
```

OUTPUT

1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16 17 18 19 20

2 marks
1.5 marks if tab is not considered

Question #3: **08**

Write a C++ program to input a 3-digit number from the user and print it into words. Use only switch structure(s) to print a string for a digit.
Note that the user will always input number according to the followings.

- ones number between 3 to 7
- tens number between 2 to 9
- hundreds number between 1 to 5

Sample Run 1:	**Sample Run 2:**
`Input number: 123` `Output: One hundred twenty three`	`Input number: 577` `Output: Five hundred seventy seven`


```
#include<iostream>
using namespace std;
int main(){
int num, num1, num2, num3, num4;
cin >> num;
num4 = num;
num1 = num % 10;
num = num / 10;
num2 = num % 10;
num = num / 10;
num3 = num % 10;
//2.5 marks

//5.5 marks for remaining three switches
//if some one have written only 1 correct switch statement then I have given him 2 marks  for that code

switch (num3){
case 1:
cout << "One hundred ";
break;
case 2:
cout << "Two hundred ";
break;
case 3:
cout << "Three hundred ";
break;
case 4:
cout << "Four  hundred ";
break;
case 5:
cout << "Five hundred ";
break;
default:
cout << "Invalide number ";
}
```

```
switch (num2){
case 2:
cout << "Twenty ";
break;
case 3:
cout << "Thirty ";
break;
case 4:
cout << "Fourty ";
break;
case 5:
cout << "Fifty ";
break;
case 6:
cout << "Sixty ";
break;
case 7:
cout << "Seventy ";
break;
case 8:
cout << "Eighty ";
break;
case 9:
cout << "Ninty ";
break;
default:
cout << "Invalide number ";
}
switch (num1){
case 3:
cout << "Three";
break;
case 4:
cout << "Four";
break;
case 5:
cout << "Four";
break;
case 6:
cout << "Six";
break;
case 7:
cout << "Seven";
break;
default:
cout << "Invalide number ";
}
/*}
else{
cout << "Invalid number";
}*/
cout << endl;
system("pause");
```

```
return 0;
}
```

Question #4: **11**

Two players are playing a game of luck. Both spin a lucky wheel twice in one turn which contains 20 numbers from 10 to 29. Each player has five turns in total (i.e., will spin the wheel 10 times). A player can win a jackpot by scoring 50 in a single turn (i.e., the sum of his/her two spins equals to 50) and the game will be over immediately. However, both players may also win a jackpot in the same turn (but, both players will take their turns no matter if the first one has hit a jackpot or not). After each turn, the scores of both players will be compared and a count of higher scorer will be maintained. In the end, if none of the players has hit a jackpot, the counts of higher scorer will be compared, and a winner will be declared based on who has scored higher in more turns than the other one. Otherwise, if both have scored higher than each other equal number of times, then the sum of the scores from all the turns will be compared for declaring a winner. If the sum matches again then the game will be declared a tie.

Hint: Use random number and sentinel.
Note: Do not use break or continue

Sample Run 1:

```
Player 1 Turn 1: Sum of two Spins = 25
Player 2 Turn 1: Sum of two spins = 30

Player 1 Turn 2: Sum of two Spins = 32
Player 2 Turn 2: Sum of two spins = 22

Player 1 Turn 3: Sum of two Spins = 29
Player 2 Turn 3: Sum of two spins = 23

Player 1 Turn 4: Sum of two Spins = 40
Player 2 Turn 4: Sum of two spins = 55

Player 1 Turn 5: Sum of two Spins = 52
Player 2 Turn 5: Sum of two spins = 46
```

Output:

```
Player 1 won as he scored more times
higher score than Player 2
```

Sample Run 2:

```
Player 1 Turn 1: Sum of two Spins = 33
Player 2 Turn 1: Sum of two spins = 26

Player 1 Turn 2: Sum of two Spins = 49
Player 2 Turn 2: Sum of two spins = 37

Player 1 Turn 3: Sum of two Spins = 29
Player 2 Turn 3: Sum of two spins = 50
```

Output:

```
The game ends as Player 2 has hit a
jackpot
```

```
#include "stdafx.h"
#include "iostream"
#include "cstdlib"
#include "ctime"
using namespace std;
int main()
{
	int p1_Turn_Score = 0, p2_Turn_Score = 0;
	int p1_high = 0, p2_high = 0;
	int p1_total = 0, p2_total = 0;
	int count;
	srand(time(0));
	for (int count = 0; count < 5 && p1_Turn_Score != 50 && p2_Turn_Score != 
50; ++count)
	{
		p1_Turn_Score = rand() % 20 + 10;
		p1_Turn_Score += rand() % 20 + 10;
		cout << p1_Turn_Score;
		p2_Turn_Score = rand() % 20 + 10;
		p2_Turn_Score += rand() % 20 + 10;
		cout << "\t" << p2_Turn_Score << endl;
		p1_total += p1_Turn_Score;
		p2_total += p2_Turn_Score;
		if (p1_Turn_Score > p2_Turn_Score)
			p1_high++;
		else if (p2_Turn_Score > p2_Turn_Score)
			p2_high++;
	}
	if (p1_Turn_Score == 50 || p2_Turn_Score == 50)
	{
		if (p1_Turn_Score == 50)
			cout << "Game ended because Player_1 scored a Jackpot\n";
		if (p2_Turn_Score == 50)
			cout << "Game ended because Player_2 scored a Jackpot\n";
	}
	else if (p1_high != p2_high)
	{
		if (p1_high > p2_high)
			cout << "Player_1 has won due to more high scores\n" ;
		else
			cout << "Player_2 has won due to more high scores\n" ;
	}
	else
	{
		if (p1_total > p2_total)
			cout << "Player_1 has won by total score\n";
		else if (p2_total > p1_total)
			cout << "Player_1 has won by total score\n";
		else
			cout << "Game is a tie";
	}

    return 0;
}
```

Question #5: **11**

A deck of cards contains 52 cards in total, distributed into 4 suits namely Heart ♥, Diamond ♦, Club ♣ and Spade ♠. Each suit has the following 13 cards: 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, King, and Ace. A player will draw cards randomly from a deck of cards and will earn 5 points on each numbered card (i.e., cards with numbered values from 2 to 10) and 10 points on each face-value card (i.e., A, K, Q or J). The game will end if a Jack card is drawn immediately after an Ace card. In that case, the last Jack card will not contribute anything to the score.

Write a C++ program to simulate this game and display the total number of cards draw and points earned on game termination. Be sure to allow uppercase as well as lowercase letters as input. An ASCII chart is given below for your reference. The program should also terminate immediately if an incorrect value for card is entered by the user. Display an appropriate message to notify the user about program termination status (see sample outputs below).

Two game samples are given below for your understanding but please DO NOT HARD-CODE for either of the scenario or otherwise. You are also not allowed to use break/continue statements in this problem for any purpose. Note that in the first sample input, T represents the number card 10 (you may use any other character of your choice).

Sample Run 1: `T 5 A J`

Output:

```
Total cards draw = 4
Total points = 20
Program terminated on getting a combination of Ace and Jack cards
```

Sample Run 2: `4 j 6 A k B`

Output:

```
Total cards draw = 5
Total points = 40
Program terminated on wrong input
```

Character	ASCII
a	97
b	98
c	99
d	100
e	101
f	102
g	103
h	104
i	105
j	106
k	107
l	108
m	109

Character	ASCII
n	110
o	111
p	112
q	113
r	114
s	115
t	116
u	117
v	118
w	119
x	120
y	121
z	122

Character	ASCII
A	65
B	66
C	67
D	68
E	69
F	70
G	71
H	72
I	73
J	74
K	75
L	76
M	77

Character	ASCII
N	78
O	79
P	80
Q	81
R	82
S	83
T	84
U	85
V	86
W	87
X	88
Y	89
Z	90

Character	ASCII
0	48
1	49
2	50
3	51
4	52
5	53
6	54
7	55
8	56
9	57

Q5 (solution)

```
char card;
bool aceJack = false, aceCard=false, faceCard=false, cardValid=false;
int totalCardDraw=0, totalPoints=0;

//input card
cin>> card;
//validate the input card
if((card>='2' && card<='9') || card=='T' || card=='t')      //numbered card
   cardValid = true;
else if(card =='A' || card=='a' || card =='K' || card=='k' || card =='Q' ||
card=='q' || card =='J' || card=='j') {                      // face-card
   cardValid = true;
   faceCard = true;
}
/*build the game logic: keep taking a card as input and stop if either
condition is met */
while(cardValid && !aceJack) {
   totalCardDraw++;
   if(!faceCard)
      totalPoints +=5;
   else if(aceCard && (card=='J' || card=='j'))
      aceJack = true;
   else {
      totalPoints +=10;
      aceCard = false;
      if(card=='A' || card=='a')
         aceCard = true;
   }
   cin>>card;
   cardValid = false;
   faceCard=false;
   //input card validation code here again …
   if((card>='2' && card<='9') || card=='T' || card=='t')    //numbered card
      cardValid = true;
   else if(card =='A' || card=='a' || card =='K' || card=='k' || card =='Q'
|| card=='q' || card =='J' || card=='j') {                   // face-card
      cardValid = true;
      faceCard = true;
   }
}

cout<<"Total cards draw:"<<totalCardDraw<<endl;
cout<<"Total points:"<<totalPoints<<endl;

//determine why the game terminated and display an appropriate message
if(!cardValid)
   cout<<"Program terminated on wrong input";
else if(aceJack)
 cout<<"Program terminated on getting a combination of Ace and Jack cards";
```

+2 (input card & validation)

+2 (correct game termination conditions)

+1 (correct scores updation on numbered card)

+2 (detecting Ace-Jack cards combination)

+2 (correct scores updation on face-card & remembering Ace card for next turn)

+2 (displaying appropriate messages on game termination)