CS118 Programming Fundamentals

Monday, November 04, 2019

Course Instructor

Rizwan Ul Haq, Zain Iqbal
Tahir Farooq, Ebad Majeed

Serial No:
Mid Term Exam
Total Time: 60 Min
Total Marks: 45

_____
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 Seven (7) 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.
7. **Use of Calculator is not allowed.**
8. 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

Question	1	2	3	4	5	Total
Points	12	10	5	8	10	45
Score						

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

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

Question No. 1	**(12)**

The Babylonian algorithm to compute the square root of a number n is as follows:

1. Make a guess at the answer (you can pick n/2 as your initial guess).
2. Compute r = n / guess
3. Set guess = (guess + r) / 2
4. Go back to step 2 for as many iterations as necessary.

The more that steps 2 and 3 are repeated, the closer guess will become to the square root of n. Write a program that inputs an integer for n, iterates through the Babylonian algorithm until guess is within 1% of the previous guess and outputs the answer as a double.

```cpp
#include <iostream>
using namespace std;
int main ()
{
int n;
double r, iGuess, nGuess;

cout<<"Please enter a positive integer: "<<endl;
cin>>n;
iGuess=(double)n/2; //value of initial 'guess'
cout<<"Your 'guess' number is "<<iGuess<<endl;
nGuess=iGuess;
do
{
r=(double)n/nGuess;
nGuess=(double)( nGuess+r)/2;
}while (nGuess>(0.0001+(( nGuess+(double) n/nGuess) / 2.0)));

cout<<"The square root of "<<n<<" is: "<<nGuess<<endl;

system("pause");
return 0;
}
```

National University of Computer and Emerging Sciences

Department of Computer Science   Chiniot-Faisalabad Campus

Question No. 2	(0.5 * 20 = 10)

Examine the C++ program below and answer the questions about that program in the blanks provided. No points will be given for overwriting and/or cutting.

```cpp
#include <iostream>
using namespace std;
int f( int, int, int&);
int g( int, int, int&);
int h( int, int, int&);
int i(int, int, int&);
int j(int , int, int&);
int k(int, int, int&);
int l(int, int, int&);
int m(int, int, int&);
int n(int, int, int&);
int o(int, int, int&);
int p(int, int, int&);

void main()
{
  int A = 2; int B = 1, int C = 0;
  cout << endl << f( A, B, C) << endl;        // Value of f: _____20_____

  cout << "Value of A: " << A << endl;         // Value of A:_____2_____

  cout << "Value of B: " << B << endl;         // Value of B:_____1_____

  A = 5; C = 0;
  cout << endl << f( A, B, C) << endl;        // Value of f: _____50_____

  cout << "Value of A: " << A << endl;         // Value of A:_____5_____

  cout << "Value of B: " << B << endl;         // Value of B:_____1_____

  A = 10; C=0;
  cout << endl << f( A, B, C) << endl;        // Value of f: _____100____

  cout << "Value of A: " << A << endl;         // Value of A:_____10_____

  cout << "Value of B: " << B << endl;         // Value of B:_____1_____

}
int f( int X, int Y, int& Z)
{
   return g(X, Y+1, Z=Z+X);                    //Value of Z: __2,5,10_____
}
int g( int X, int Y, int& Z)
{
   return h (X, Y+1, Z=Z+X);                   //Value of Z: __4,10,20_____
}
int h( int X, int Y, int& Z)
{
   return i(X, Y+1, Z=Z+X);                    //Value of Z: ___6,15,30____
}
```

```
int i( int X, int Y, int& Z)
{
    return j (X, Y+1, Z=Z+X);                    //Value of Z: ___8,20,40____
}

int j( int X, int Y, int& Z)
{
    return k (X, Y+1, Z=Z+X);                    //Value of Z: ___10,25,50___
}
int k( int X, int Y, int& Z)
{
    return l (X, Y+1, Z=Z+X);                    //Value of Z: ___12,30,60___
}
int l( int X, int Y, int& Z)
{
    return m (X, Y+1, Z=Z+X);                    //Value of Z: ___14,35,70___
}
int m( int X, int Y, int& Z)
{
    return n (X, Y+1,Z=Z+X);                     //Value of Z: ____16,40,80__
}
int n( int X, int Y, int& Z)
{
    return o (X, Y+1,Z=Z+X) ;                    //Value of Z: ____18,45,90__
}
int o( int X, int Y,  int& Z)
{
    return p(X, Y+1,Z=Z+X);                      //Value of Z: ____20,50,100_
}
int p( int X, int Y,int& Z)
{
    return Z;                                    //Value of Z: __20,50,100__

}
```
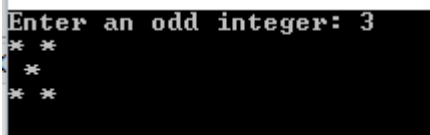
National University of Computer and Emerging Sciences

Department of Computer Science   **Chiniot-Faisalabad Campus**

Question No. 3	**(01 * 05 = 5)**

Find the error(s) in each of the following program segments, and explain how the error(s) can be corrected

<table>
<tr>
<td>

a)
```
int g()
{
cout << "Inside function g" <<
endl;
int h()
{
cout << "Inside function h" <<
endl;
}
}
```
</td>
<td>

<Solution>
int g()
{
cout << "Inside function g" << endl;
}
int h()
{
cout << "Inside function h" << endl;
}
</td>
</tr>
<tr>
<td>

b)
```
int sum( int x, int y )
{
int result;
result = x + y;
}
```
</td>
<td>

<Solution>
No return statement
Or
Function should be void
</td>
</tr>
<tr>
<td>

c) `#include <iostream>`
```
 int var = 20;
 int main()
 {
    int var = var;
 cout<<"In main value of  var"<<var
    return 0;
 }
```
</td>
<td>

<Solution>

Local and global variable same name
</td>
</tr>
<tr>
<td>

a)
```
void f( double a );
{
float a;
cout << a << endl;
}
```
</td>
<td>

<Solution>
void f(double a)
{
==float n;==
cout << a << endl;
}
</td>
</tr>
<tr>
<td>

b)
```
void product()
{
int a;
int b;
int c;
int result;
cout << "Enter three integers: ";
cin >> a >> b >> c;
result = a * b * c;
cout << "Result is " << result;
return result;
}
```
</td>
<td>

<Solution>
void product()
{
int a;
int b;
int c;
int result;
cout << "Enter three integers: ";
cin >> a >> b >> c;
result = a * b * c;
cout << "Result is " << result;
==//return result;==
}
</td>
</tr>
</table>

Question No. 4	02*04 = 08

Dry Run following lines and mention output of each program segment in the table.

a)
``cpp
for (int i = 1; i <= 5; ++i)
{
 for (int j = 1; j <= 3; ++j)
 {
   for (int k = 1; k <= 4; ++k)
 cout << '*';
 	
 cout << endl;
 } // end inner for
cout << endl;
} //
``
	<u>Your Output here</u>
	``
	****
	****
	****
	
	****
	****
	****
	
	****
	****
	****
	
	****
	****
	****
	
	****
	****
	****
	``

b) for loop to display following pattern
``
*
**
***
****
*****
``
	<u>Your Output here</u>
	``cpp
	for (i = 1; i <= 5 ; i++)
	            {
	                    for (j = 1; j <= i; j++)
	                        cout << "*";
	                    cout << endl;
	            }
	``

c)
``cpp
int a, c, t;
cout<<"Enter an odd integer: "; cin>>t;
 	
for(a=1; a<=t; a++)
{
for(c=1; c<=t; c++)
{
if(c==a || c==(t+1-a))
cout<<"*";
else
cout<<" ";
}
cout<<endl;
}
``
	<u>Identify the output if user enters 3</u>
	``
	Enter an odd integer: 3
	* *
	 *
	* *
	``

d)
``cpp
#include <iostream>
Int main() {
int n = 3;
while (n >= 0)
{
cout << n * n << endl;
--n;
}
cout << n << endl;
while (n < 4)
cout << ++n << endl;
cout << n << endl;
while (n >= 0)
cout << (n /= 2) << endl;
return 0;
``
	<u>Your Output here</u>
	``
	9
	4
	1
	0
	-1
	0
	1
	2
	3
	4
	4
	``
	2
	
	0...........

```
}
```

Question No. 5	(10)

The simple program Fibonacci number does not check whether the first number entered by the user is less than or equal to the second number and whether both the numbers are nonnegative. Also, the program does not check whether the user entered a valid value for the position of the desired number in the Fibonacci sequence. Rewrite that program so that it checks for these things.