Practice for exams

For output questions, make sure to not use compiler. Dry run on paper to check the output. In exams, you might need to design a tracetable in such questions so you must need to practice how to dry run any code w/o compiler.

Assume that the purpose of the program is to add up all the odd numbers between 0 and 100. Fill in the blanks:

```
int n;
int sum = 0;
for (...; ...; ...)
{
sum += n;
}
```

a) n = 1; n + 2; n <= 100
b) n = 1; n <= 100; n + 2
c) n = 1; n <= 100; n += 2
d) n = 1; n <= 100; n++ 2
e) b or c

Given the following definitions, what is the value of b[1][0]?

int b[2][2] = { { 1 }, { 3 , 4 } };

(1). 0

(2). 1

(3). 3

(4). this isn't a valid definition

Assume memory address and write down the output of the following code:

```
#include <iostream>

using namespace std;

int main()

{

        int var1 = 10, var2 = 5;

        int* p1, * q1, * r1;

        int** g;

        int var3;
        var3 = ++var2;
        p1 = &var1;
        q1 = &var2;
        r1 = &var3;
        g = &r1;
        *p1 = var3++;
        *q1 = ++var1;
        Cout<<*p1<<" "<<p1<<" "<<&p1<<endl;
        Cout<<*q1<<" "<<q1<<" "<<&q1<<endl;
        Cout<<*g<<" "<<g<<" "<<&g<<endl;
        return 0;
}
```

A function prototype can always be omitted when:

(1).a function is defined before it is first invoked

(2).a function is invoked before it is first defined

(3).a function takes no arguments

(4).a function does not return a value

Write statement:

(1).that assigns a random integer to the variable n in the range: $100 \leq n \leq 112$

(2).that assigns a number at random to the variable n, from the set [3,5,7,9,11]

(3).that copies array a in to the first portion of array b. Assume double a[11],b[34];

(4).that inputs a value to element 4 of single-subscripted floating point array b

---

What does the following program do?

```
#include <isostream>
int mystery (int a, int b);
int main(void)
{
```

```
int x;

int y;

cout<<“enter two integers:”;

cin>>x>> y;

cout<<mystery(x,y);

return 0;

}

int mystery (int a, int b)

{

 if (b==1) {

 return a;

 }

 else {

 return a+ mystery(a,b-1);

 }

}
```

Write a program that simulates coin tossing. For each toss of the coin, the program should print Heads or Tail Write a program that simulates coin tossing. For each toss of the coin, the program should print Heads or Tails. Let the program toss the coin 100 times and count the number of times each side of the coin appears. Print the results. The program should call a separate function flip that takes no arguments and return 0 for tails and 1 for heads

---

Give the function header for each of the following functions:

a) Function hypotenuse that takes two double-precision, floating-point arguments, side1 and side2, and returns a double-precision, floating-point result.
b) Function smallest that takes three integers, x, y and z, and returns an integer.
c) Function instructions that does not receive any arguments and does not return a value.
d) Function intToDouble that takes an integer argument, number, and returns a doubleprecision,
floating-point result.

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

```
a) int sum( int n )
{
        if ( 0 == n )
                return 0;
        else
                n + sum( n - 1 );
}
b) void f( double a );
{
        float a;
        cout << a << endl;
}
c) void product()
{
        int a = 0;
        int b = 0;
        int c = 0;
        cout << "Enter three integers: ";
        cin >> a >> b >> c;
        int result = a * b * c;
        cout << "Result is " << result;
        return result;
}
```

Function floor can be used to round a number to a specific decimal place. The statement
y = floor(x * 10 + 0.5) / 10;
rounds x to the tenths position (the first position to the right of the decimal point). The statement
y = floor(x * 100 + 0.5) / 100;
rounds x to the hundredths position (the second position to the right of the decimal point). Write a program that defines four functions to round a number x in various ways:
a) roundToInteger(number)
b) roundToTenths(number)
c) roundToHundredths(number)
d) roundToThousandths(number)
For each value read, your program should print the original value, the number rounded to the nearest integer, the number rounded to the nearest tenth, the number rounded to the nearest hundredth and the number rounded to the nearest thousandth.

Write a function multiple that determines for a pair of integers whether the second is a multiple of the first. The function should take two integer arguments and return true if the second is a multiple of the first, false otherwise. Use this function in a program that inputs a series of pairs of integers.

Consider the following type definition:

```
struct ShoeType
{
        char style;
        double price;
};
```

Given this structure type definition, what will be the output produced by the following code?

```
ShoeType shoe1, shoe2;
shoe1.style ='A';
shoe1.price = 9.99;
cout << shoe1.style << " $" << shoe1.price << endl;
shoe2 = shoe1;
shoe2.price = shoe2.price/9;
cout << shoe2.style << " $" << shoe2.price << endl;
```

Write a definition for a structure type for records consisting of a person's wage rate, accrued vacation (which is some whole number of days), and status (which is either hourly or salaried). Represent the status as one of the two *char* values 'H' and 'S'. Call the type EmployeeRecord.

Design an application SE-StudentMgtSystem that stores the record of a student using a struct. Define a struct consisting of name of student, quiz marks, midterm marks, and final term marks. The number of students whose record needs to be stored is fixed, i.e 10. You might need to create an array of structs. You are required to do the following:

1) Assume you have a file "studentsRecord.txt" that has record of 10 students (one record each line). The individual student's record consists of name of the student, quiz marks, midterm and finalterm marks, all separated by a space. Read the contents of file and initialize an array of structs with each student's record.

2) Implement a function "calGrade" that takes a struct as an input parameter. This function should then read the contents of struct and return the percentage result using the formula:

result=(quiz+midterm+final)/220 x 100

Write output of the following code:

```
#include <iostream>
#include <string>
using namespace std;
void Average(int, int);
void Message(void);
void func(int, int*);
int fun2(int*);
int fun3();
int main()
{
        int num1=30, num2= 40;
        Message(); // Line 4
        func(num1, &num2);
        Average(num1, num2); // Line 5
        cout<<"Number 1 is: "<<num1<<endl<<"Number 2 is : "<<num2<<endl;
        return 0;
}
void func(int num1, int* num)
{
       int num2= *num;
         num1= num1+ num2 / 20;
         num1 - = 10;
         num2 - = 20;
       cout<<"Number is now: "<<num1<<endl;
        num1= fun2(&num1);
        cout<<num1<<"  "<<num<<"  "<<num2<<endl;
}
int fun2(int * num)
{
        *num= *num +10;
          cout<<"Hello I am fun2"<<endl;
          int temp= fun3();
          Message();
          cout<<"Hello I am fun2"<<endl;
          return temp;
}
int fun3()
{
        return 50;
}
void Average(int x, int y)
{
        cout<<"(x + y)/2.0 = ”<< (x+y)/2.0);
}
void Message(void)    {
         cout<<”Average “;
}
```

Write output of the following code:

```
    int x[]={1,4,8,3,6,9,5,2};

    int *p=&x[5];

    for (int i=0;p<&x[2];p--)

          cout<<*p;
```

```
int *q=&x[7];

for ( ;p<q;p++,q--)

        cout<<*p<<*q;

p=new int[6];
p= &x[7];

for (i=0;i<7;i++,p--)

        *p=x[i];

for (p=&x[7];i>2;i--,p++)

        cout<<*p<<x[i];
```

Write a program that :

- **Ask user to enter size and elements of 2D matrix.**
- **Ask user for row or column rotation.**
- **Ask number of steps of rotation.**
- **Perform rotation .**

```
Sample:
Enter number of rows: 2
Enter number of Columns: 3
Enter Elements:
1 2 3 4 5 6
Press 1 for row operation and 2 for column operation: 1
Enter row number: 1
How many steps for rotation: 2

Resultant Matrix after rotation:
2 3 1
4 5 6
```

Note: Implements proper checks for invalid inputs.

Identify the scope of the mentioned variables and fill in the table.

Line#	
1	float a = 1;
2	int main()
3	{
4	int j=9;
5	cout<< “j: ”<<j;
6	my_func(j);
7	return 1;
8	}
9	void my_func(int x)
10	{
11	float b = 77;
12	cout<< “b: ”<<b;
13	}
14	void setup()
15	{
16	float d = 2;
17	d=d*12;
18	cout<< “d: ”<<d;
19	}

Variable name	Scope		Variable type (Local/ Global)
	From Line#	To Line#	
a			
j			
b			
d			
x			