National University of Computer and Emerging Sciences, Lahore Campus

Course Name:	Object Oriented Programming	Course Code:	CS1004
Degree Program:	BS (CS, SE, DS, BS Robotics)	Semester:	Summer 2023
Exam Duration:	180 Minutes	Total Marks:	70
Paper Date:	9-August-2023	Weight	45
Section:	ALL	Page(s):	10
Exam Type:	Final Exam		

Student : Name:_____ **Roll No.**_____ **Section:**_____

Instruction/Notes: Attempt all questions. Answer in the space provided. **Answers written on rough sheet will not be attached and marked**. Do not use pencil or red ink to answer the questions**.** In case of confusion or ambiguity make a reasonable assumption. Properly comment your code.

Question 1: (CLO: 1) **(Marks: 10)**

Determine output for the code segment given below. There is no syntax error in the code.

```
#include<iostream>
using namespace std;
class A
{
private:
	int a;
public:
	A(int x = 10) { a = x; cout << "A() called.\n"; }
	~A() { cout << "~A() called for a = " << a << endl; }
	void Print() { cout << "a = " << a << endl; }
};
class B
{
private:
	int b;
	A a;
	A* aptr;

public:
	B(){ b = 0; aptr = 0; cout << "B() called." << endl; }
	B::B(int x) :a(x-2), b(x), aptr(0)
	{
		cout << "B() called for b = " << b << endl;
	}
	void Print() {
		cout << "b = " << b << endl; a.Print();
		if (aptr != 0) aptr->Print();
	}
	~B(){ cout << "~B() called for b = " << b << endl; }
};
void main()
{
	B b1(3);
	cout << "-------------------------\n";
	b1.Print();
}
```

OUTPUT:

Question 2: (CLO: 4) **(Marks: 10)**

Determine output for the code given below. There is no syntax error in the code.

```
#include<iostream>
using namespace std;
char* generateA(char* p, char*& q)
{
	p = new char;
	*p = *q + 1;
	*q = *p + 2;
	return p;
}
char* generateB(char*& p, char* q)
{
	q = new char;
	*q = *p + 3;
	*p = *q + 4;
	q = generateA(p, q);
	return q;
}
int main()
{
	char x = 'A';
	char* ptr1 = &x;
	char* ptr2 = new char;
	*ptr2 = 'N';
	cout << *ptr1 << " " << *ptr2 << endl;
	ptr1 = generateB(ptr1, ptr2);
	cout << *ptr1 << " " << *ptr2 << endl;
	ptr2 = generateA(ptr1, ptr2);
	cout << *ptr1 << " " << *ptr2;
	delete ptr2;
	return 0;
}
```

OUTPUT:

Question 3: (CLO: 2) **(Marks: 20)**

Write user defined functions to accomplish the following task:

Creating a 2D jagged array from a two-dimensional array A, with size N rows and M columns such that it will store bold/highlighted part of the array.

e.g,

Input Array A looks like this:

2	**3**	**1**	**5**	**0**
7	**1**	**5**	**3**	**1**
2	5	**7**	**8**	**1**
0	1	5	**0**	**1**
3	4	9	1	**5**

Resultant jagged array will look this:

2	3	1	5	0
1	5	3	1	
7	8	1		
0	1			
5				

You have to write three functions:

1. Function 1 for creating jagged array in the memory using pointers
   int AllocatingMemory(int** arr, int rows, int cols)**
2. Function 2 to copy the elements of the arr to the resultant jagged array that is created by Function 1
   void FillResultantJaggedArray(…) //decide arguments by yourself
3. Function 3 to display elements of the resultant jagged array that is filled by Function 2
   void PrintResultantJaggedArray(…)//decide arguments by yourself

Note:

You need to decide the prototype of the functions very carefully so that it will work fine. You don't need to write main function (No credit will be given).

Question 4: (CLO: 3) **(Marks: 20)**

```
class circle
{
  public:
     void print() const;
     void setRadius(double);
     void setCenter(double, double);
     void getCenter(double&, double&);
     double getRadius();
     double areaOfCenter();
     circle();
     circle(double, double, double);
  private:
      double xCoordinate;
      double yCoordinate;
      double radius;
};
```

```
class cylinder: public circle
{
   public:
        void print() const;
        void setHeight(double);
        double getHeight();
        double volume();
        double areaOfCylinder();
        cylinder();
        cylinder(double, double, double,
double);
    private:
         double height;
};
```

Formulas:

Area of circle= $2\pi r^2$

Area of cylinder= $2\pi rh$ + Area of circle

Volume of cylinder= Area of cylinder * height

Program should work with the given main:

```
int main() {
   circle* cyPtr = new cylinder(2.0, 1.0, 3.0, 5.0)    //Center:2.0 and 1.0---Radius:3.0--- Height:5.0
   cout << "Cylinder Information:" << endl;
   cyPtr->print();
   cout << "Area: " << cyPtr->area() << endl;
   cout << "Volume: " << cyPtr->volume() << endl;
   delete cyPtr;
   return 0;
}
```

Write the definitions of the member functions of the classes circle and cylinder. Identify the member functions of the class cylinder that overrides the member functions of the class circle. **Please ensure that the functions are executing correctly. If any components are absent, kindly populate and finalize them.**

Question 5: (CLO: 5) **(Marks: 10)**

Determine output for the code given below. Explicitly mention if the program crashes or leaks any resources. There is no syntax error in the code.

```
#include<iostream>
using namespace std;
class ATM
{
	int cash;
	static int tr_Count;
	static int tr_Attempt;
public:
	ATM(int c)
	{
		cash = c;
	}
	void withdraw(int amount)
	{
		if (amount > cash)
		{
			tr_Attempt++;
			cout << "Error: 404\n";
			char e[] = { "Error: 404" };
			throw e;
		}
		else if (amount < 500)
		{
			tr_Attempt++;
			cout << "Error 401\n";
			throw amount;
		}
		else if (amount % 500 != 0)
		{
			tr_Attempt++;
			cout << "Amount should be the mutiple of 500\n";
			throw 'e';
		}
		cash -= amount;
		cout << "Remaining cash in ATM " << cash << endl;
		tr_Count++;
	}
	int checkCash()
	{
		return cash;
	}
	static int getTrAttempt()
	{
		return tr_Attempt;
	}
	static int getTrCount()
	{
		return tr_Count;
	}
};
int ATM::tr_Count = 0;
int ATM::tr_Attempt = 0;

int main()
{
	int w_Am[10] = { 3000, 000, 5000, 500, 200, 1000, 800, 500 };
	int am;
	ATM a1(5000);
	int k = 0, i;
```

```
while (a1.getTrAttempt() < 3)
	{
		try
		{
			a1.withdraw(w_Am[k]);
		}
		catch (char* s)
		{
			am = a1.checkCash();
			k++;
			while (w_Am[k] > am)
			{
				k++;
			}
			a1.withdraw(w_Am[k]);
		}
		catch (int e)
		{
			am = a1.checkCash();
			k++;
			while (w_Am[k] > am || w_Am[k] < 500)
			{
				k++;
			}
			a1.withdraw(w_Am[k]);
		}
		catch (char excp)
		{
			while (w_Am[k] % 500 != 0)
			{
				k++;
			}
			a1.withdraw(w_Am[k]);
		}
		k++;
	}
	system("pause");
}
```

OUTPUT: