National University of Computer and Emerging Sciences

Lahore Campus

Object Oriented Programming (CS1004)

Date: 22-05-2025

Course Instructor(s)

SK, SI, UA, UN, NA, AK, UA, RD

Final Exam

Total Time (Hrs): 3

Total Marks: 60

Total Questions: 5

____________ Roll No   ____________ Section   ____________________ Student Signature

Attempt all the questions in the space provided on question paper except Question 1 and 2. You can use rough sheet but do not attach it with the question paper. Do not use red ink or pencil to solve the exam. Extra code will result in negative marks. Added code should be neat and clean.

CLO #3: Model an algorithmic solution for a given problem using OOP

Question 1: (Movie Recommender system) [5+10 Marks]

Suppose you are working as an intern for IMDB (Internet Movie DataBase) and your boss has decided to use user ratings for movie recommendation algorithm. To store the ratings it is decided to design a class that ratings using user id and movie id. The algorithm needs the following information: 1) List of movies rated by for each user and 2) list of users who rated a particular movie. To efficiently process this information it is decided that the rating class must have two 2-dimensional dynamic arrays users and movies. The users array must have number of rows equal to number of users in the system and movies array must have number of rows equal to number of movies in the system. Each index of users array must store the ids of the movies he/she has rated. Similarly each index of movies array must store the ids of users who have watched that movie. Consider the following example: Let there are four movies {1, 2, 3, 4} and four users in the system {A, B, C, D}. Below are the interactions:

- User A watched Movie 1 and 3
- User B watched Movie 2 and 4
- User C watched Movie 1, 2 and 3
- User D watched Movie 4 only

Below are the sample 2-dimensional arrays of users and movies and partial definition of class ratings. Please assume that each movie has an int ID and each user has a char ID. Also assume that the function get_user_index returns the index of the array where data of that user is stored. For example given char 'A' this function will return 0 and given 'C' it will return 2.

Users array

A	1	3	-1	
B	2	4	-1	
C	1	2	3	-1
D	4	-1		

Movies array

1	A	C	!
2	B	C	!
3	A	C	!
4	B	D	!

```
class ratings{
    int** users;//2_D array to store int movie IDs followed by -1 for each user
    char** movies;// 2-D array to store char IDs of users followed by !
    int total_users;
    int total_movies;
    int get_user_index(char id)//returns the array index corresponding to char user ID
    //rest of the class implementation };
```

i. Define a function Add_rating (char u, int m) that adds a rating to the object. For example if user 'D' watches a new movie 1, then the Users and Movies array must be updated as follows. Make sure to properly manage the dynamic memory.

Users array

A	1	3	-1	
B	2	4	-1	
C	1	2	3	-1
D	4	1	-1	

Movies array

1	A	C	D	!
2	B	C	!	
3	A	C	!	
4	B	D	!	

```
// Function to add a rating
    void Add_rating(char user_id, int movie_id) {
        int user_index = get_user_index(user_id);

        // Add movie to user's list
        int i = 0;
        while (users[user_index][i] != -1) i++;
       int * temp1 = new int[i+1];
       for(i=0;users[user_index][i]!=-1;i++)
            temp1[i] = users[user_index][i];
        temp1[user_index][i] = movie_id;
        temp1[user_index][i + 1] = -1; // Maintain termination
        delete [] users[user_index];
        users[user_index] = temp1;

        // Add user to movie's list
        int movie_index = movie_id - 1;
        int j = 0;
        while (movies[movie_index][j] != '!') j++;
        char * temp2 = new char[i+1];
       for(i=0;users[movie_index][i]!=-1;i++)
            temp2[i] = users[movie_index][i];
        temp2[movie_index][i] = user_id;
        temp2[movie_index][i + 1] = !; // Maintain termination
        delete [] movies[movie_index];
        movies[movie_index] = temp2;
    }
```

ii. Define a function recommend that takes a char user ID u as parameter and returns a dynamic integer array of all the movies recommended for user u. A movie m will be recommended to use u if u has not already watched m and there exists a user v such that user v has watched movie m and user u and v have watched at least one common movie. For example user 'A' will be recommended movie 2 because user C has watched movie 2 and user 'A' and user 'C' has watched 2 common movies {1, 3}. Similarly use 'A' will be recommended movie 4 because user 'D' has watched movie 4 and user 'A' and user 'D' has watched 1 common movie {1}. So in this example your function must return {2, 4}

```
template<class T>
bool search(T * arr, int size, T key){
   for(int i=0;i<size;i++){
        if(arr[i] == key)
     return true;
   }
   return false;
}
 // Function to recommend movies for a user
   int* recommend(char user_id, int& recommended_count) {
   int user_index = get_user_index(user_id);
   int watch_count =0;
//count number of movies wateched by given user_id
while(users[user_index][watch_count]!=-1) watch_count++;
 // Find users who share common movies with the given user_id
      char similar_users=new char[total_users];
      int k=0;
      for (int j = 0 : users[user_index][j]!=-1;j++) {
         int movie = users[user_index][j];
         int movie_index = movie - 1;
         for (int i = 0; movies[movie_index][i] != '!'; i++) {
            if (movies[movie_index][i] != user_id&& !search(similiar_users, k, movies[movie_index][i])) {
               similar_users[k++] = movies[movie_index][i];
            }
         }
      }
     int recommended_movies= new int[total_movies];
     int count = 0;
      // Find movies watched by similar users but not by the given user
      for (int j=0; j<k;j++) {
         int similar_user_index = get_user_index(similar_users[j]);
         for (int i = 0; users[similar_user_index][i] != -1; i++) {
            int movie_id = users[similar_user_index][i];
            if (!search(recommended_movies, count, movie_id) &&
               !search(users[user_index],watch_count,movie_id)) {
                recommended_movies[count++] = movie_id;
            }
         }
      }

      int* recommendations = new int[count];
      for (int i = 0; i < count; i++) {
         recommendations[i] = recommended_movies[i];
      }
      delete [] similar_users;
      delete []recommended_movies’
      return recommendations;
   }
```

CLO#2: Apply OOP concepts (Encapsulation, Inheritance, Polymorphism,Abstraction) to computing problems for the related program

Question 2: [15 Marks]
We are designing a basic **Exam Management System** that handles a **Question Paper** composed of multiple types of **questions**. Each question in the paper can either be an **Essay Question** or a **Multiple Choice Question (MCQ)**. Every Question has a *QuestionText (string)* and *marks (int)*. An EssayQuestion has a *word_limit (int)*. Every MCQ can have *options(string*)*, *optionsCount(int)* and *correctOption(string)*. **Your task is to write C++ classes in a proper hierarchy which enables the driver given below to compile and produce the given output.** *Your code will be marked for best using concepts studied in class. You can freely use built-in string functions. You can freely choose between string or c-string to save corresponding data. There should not be any memory leakage or exception in your code.*

Driver Program:

```
void main() {
    QuestionPaper paper; /*For now, a question paper can have maximum 10 questions
(we are not supporting dynamic grow/shrink).*/

    Question* q1 = new EssayQuestion("Explain the concept of inheritance in OOP.",
10, 300);//10 Marks, word limit 300
      paper.addQuestion(q1);

      string optionsForQ2[4] = {"Inheritance", "Encapsulation", "Virtual
Functions", "Abstraction"};
    Question* q2 = new MCQQuestion( "Which of the following is a polymorphic feature
in OOP?", 5,optionsForQ2,4,"Virtual Functions"); /*5 Marks, 4 options for this MCQ*/
      /*An MCQ can have five options at max and user will always add atleast two
option, you don't have to check boundary conditions, dynamic grow/shrink also not
required*/
    paper.addQuestion(q2);
    paper.showAllQuestions();/* must call functions of the Question class or its
derived classes*/
}
```

Required Output:

```
Question [10 Makrs]: Explain the concept of inheritance in OOP.
Word Limit: 300

Question [5 Makrs]: Which of the following is a polymorphic feature in OOP?
1. Inheritance
2. Encapsulation
3. Virtual Functions
4. Abstraction
(Correct Option: Virtual Functions)
```

Solution:

```
#include <iostream>
#include <string>
using namespace std;

class Question {
protected:
    string questionText;
    int marks;

public:
    Question(string text, int m) : questionText(text), marks(m) {}
       virtual void display()
       {
              cout << "Question ["<< marks<<" Makrs]: " << questionText << endl;
       }
    virtual ~Question() {}
};

class EssayQuestion : public Question {
private:
    int wordLimit;

public:
    EssayQuestion(string text, int m, int limit)
        : Question(text, m), wordLimit(limit) {}

    void display(){
        Question::display();
        cout << "Word Limit: " << wordLimit << endl;
    }
};

class MCQQuestion : public Question {
private:
    string* options;
       int optionsCount;
    string correctOption;

public:
    MCQQuestion(string text, int m, string* opts, int count, string correct)
        : Question(text, m), optionsCount(count), correctOption(correct)
       {
              options = new string[count];
              for(int i=0; i< count; i++)
              {
                     options[i] = opts[i];
              }
       }

    void display(){
              Question::display();
        for (int i = 0; i < optionsCount; ++i) {
            cout << i + 1 << ". " << options[i] << endl;
        }
        cout << "(Correct Option: " << correctOption << ")" << endl;
    }
       ~MCQQuestion(){
              delete[] options;
       }
};

class QuestionPaper {
private:
    Question* questions[10];
       int totalQuestions;

public:
       QuestionPaper():totalQuestions(0){
       }
    void addQuestion(Question* q) {
              questions[totalQuestions++] = q;
    }
    void showAllQuestions() const {
              for (int i=0; i< totalQuestions ; i++) {
                     questions[i]->display();
            cout << endl;
        }
    }

    ~QuestionPaper() {
        for (int i=0; i< totalQuestions ; i++) {
                     delete questions[i];
        }
    }
};

// Demo usage
void main() {

    QuestionPaper paper; /*For now, a question paper can have maximum 10 questions (we
are not supporting dynamic grow/shrink).*/

    Question* q1 = new EssayQuestion("Explain the concept of inheritance in OOP.", 10,
300);//10 Marks, word limit 300
       paper.addQuestion(q1);

       string optionsForQ2[4] = {"Inheritance", "Encapsulation", "Virtual Functions",
"Abstraction"};
    Question* q2 = new MCQQuestion( "Which of the following is a polymorphic feature in
OOP?", 5,optionsForQ2,4,"Virtual Functions"); /*5 Marks, 4 options for this MCQ*/
       /*An MCQ can have five options at max and user will always add atleast two option,
you don’t have to check boundary conditions, dynamic grow/shrink also not required*/
       paper.addQuestion(q2);

    paper.showAllQuestions();
}
```

CLO#2: Apply OOP concepts (Encapsulation, Inheritance, Polymorphism,Abstraction) to computing problems for the related program

Question 3: [5+5=10 Marks]

a) What will be the output of the above code snippet? Justify your answer with solid reason.

```
class Animal {
public:
    virtual void speak() {
        cout << "Animal speaks" << endl;
    }
};
class Dog : public Animal {
public:
    void speak() override {
        cout << "Dog barks" << endl;
    }
};
class Cat : public Animal {
public:
    void speak() override {
        cout << "Cat meows" << endl;
```

Reason and Output:

It does not show polymorphic behavior properly. This is because even though Dog and Cat are assigned, object slicing (When you assign a derived class object to a base class object by value, only the base part of the derived object is copied the rest is &quot;sliced off.&quot; This is called object slicing.) occurs only the Animal part is stored in the array. Polymorphism always work with pointers and references. If we use pointers problem will be solved.

```
    }
};
void speakFunc(Animal a[],int size)
{
    for (int i = 0; i < size; i++) {
        a[i].speak();
    }
}
int main() {
    Animal animals[2];
    animals[0] = Dog();
    animals[1] = Cat();
    speakFunc(animals,2);
    return 0;
}
```

```
Output:
Animal speaks
Animal speaks
```

b) What will be the output of the following code snippet? Is there anything wrong? If yes then how to correct that? Give appropriate reasons

```
class X {
public:
    X() { cout << "X() "; }
    virtual void f() { cout << "X::f() "; }
    ~X() { cout << "~X() "; }
};
class Y : public X {
public:
    Y() { cout << "Y() "; }
    void f() override { cout << "Y::f() "; }
    ~Y() { cout << "~Y() "; }
};
int main() {

    X* xp = new Y[2];

    for (int i = 0; i < 2; ++i)
        xp[i].f();
    delete []xp;
    return 0;
}
```

Output:

X() Y() X() Y() Y::f() Y::f() ~X()

CLO#1: Demonstrate the basic concepts of OOP

Question 4: [5+2=7 Marks]

a) Find the output of the program after removing the errors if any.

```
class Counter {
    int value;
public:
    Counter(int v ) : value(v) {}
    Counter& operator++() {
        ++value;
        return *this; }
    void print() const {
        cout << "Value: " << value << endl; }
};
int main() {
    Counter c(0);
    ++(++c);
    (++c)++;
     c.print();
    }
```

There are errors in program as operator to overload postfix increment functionality is not implemented.
Postfix Increment code is:

```
Counter operator++(int) {
Counter temp = *this;
value++;
return temp;
}
```

The program output should be 4 now.

b) Identify the issue in the code and fix the issue if any.

```
#include <iostream>
using namespace std;
int& trickyFunction() {

      int value = 999;
      return value;

}
int main() {
   int& ref = trickyFunction();
   cout << "Shadowed Value Ref: " << ref <<
endl;
}
```

Issue and Correction:
Value Goes out of Scope.

```
#include <iostream>
using namespace std;

int& trickyFunction() {
   static int value = 999;
   return value;
}

int main() {
   int& ref = trickyFunction();
   cout << "Static Value Ref: " << ref << endl;
}
```

CLO#4 Apply good programming practices

Question 5: [8+5=13 Marks]

a) Predict the output/error(s). There is no syntax error.

```
bool flag = true;
class File {
public:
   File(const char* name) {
      if(flag == false)
         throw 'q';
/*If both strings are identical, it returns 0.
  If first string is greater it returns positive
  And return negative otherwise.
*/
      else if(!strcmp(name, "1.txt")){
         flag = false;
         throw ("file does not exist");
      }
      else
         cout << "Open File\n";
   }
   ~File() { cout << "Close File\n"; }
};
class Buffer {
public:
   Buffer(int value){
      if(value > 10){
         flag = false;
         throw 504;
      }
      cout << "Allocate Buffer\n";
   }
   ~Buffer() { cout << "Free Buffer\n"; }
};
```

```
void readHeader() {
   File* f = nullptr;
   try{
      char fileName[10];
      cin >> fileName;
      f = new File(fileName);
      cout << "Header Read\n";
   }
   catch (char e){
      cout << "can't open the file\n";
   }
   delete f;
}
void parseFile() {
   int value;
   cin >> value;
   Buffer* b = nullptr;
   try {
      b = new Buffer(value);
      readHeader();
   } catch (int exception) {
      cout << "Memory Error\n";
      readHeader();
   }
   cout << "File Parsed\n";
   delete b;
}
```

```
int main() {
    try {
        parseFile();
    }
    catch (const char* e) {
        cout << "Parsing Failed\n";
        cout << e << endl;
    }
    if(flag)
        throw false;
    cout << "continue...\n";
}
```

Case1: input (6 and “1.txt”)

Allocate buffer
Parsing failed
File does not exist
Continue…

Case2: input (12 and “3.txt”)
Memory error
Can’t open the file
File parsed
Continue…

Case3: input (4, “a.txt”)
Allocate buffer
Open file
Header read
Close file
File parsed
Free buffer
Program crashed

b) Provide the output of this code. If there is any issue fix the issue and give the output.

```
#include <iostream>
#include <cstring>
using namespace std;
template <typename T>
T maxVal(T a, T b) {
    if (a > b)
        return a;
    else
        return b;
}
```

```
int main() {
    cout << maxVal(10, 20) << endl;
    cout << maxVal(3.14, 2.71) << endl;
    char f1[] = “Apple”;
    char f2[] = “Banana”;
    cout << maxVal(f1, f2) << endl;
}
```

Output and Correction:

```
// Output: 20
// Output: 3.14
// Output: Banana

// Specialization for const char*
template &lt;&gt;
const char* maxVal(const char* a, const char* b) {
if (strcmp(a, b) &gt; 0)
return a;
else
return b;
}
```