CPSC 256 – Midterm Test 4

The test consists of two questions. The first question is worth 15 points and the second is worth 25 points, for a total of 40 points for the entire test.

You should complete your solutions on the test paper itself. You are allowed pens, pencils, erasers, and blank scrap paper. No other aids are allowed.

Question 1: Inheritance and Polymorphism (15 points total)

You are building a program to manage different types of vehicles. The base class is Vehicle, and each derived class represents a specific vehicle (e.g., Car and Bike). The Vehicle class contains a virtual method honk() that is implemented differently in each derived class.

1. (4 points)

Write the declaration of the Vehicle base class, including:

• A protected member variable for the vehicle’s name.

• A public virtual method double honk() = 0; to make the class abstract.

• An initialization constructor which sets the internal name variable to the

provided vehicle name, and a default destructor.

1. (8 points)

Implement two derived classes:

• Car: It should override the honk() method to print the phrase “Beep Beep!”

• Bike: It should override the honk() method print the phrase “Ring Ring!”

1. Explain the role the “virtual” keyword in achieving polymorphism. What will happen if honk() is NOT declared virtual in the base class? Write your explanation in the space below:

int main() {

// Create an array of Vehicle pointers

Vehicle\* vehicles[2];

vehicles[0] = new Car(); // Car object

vehicles[1] = new Bike(); // Bike object

// Demonstrate polymorphism

for (int i = 0; i < 2; i++) {

vehicles[i]->honk(); // Calls the overridden honk() method

}

// Free dynamically allocated memory

for (int i = 0; i < 2; i++) {

delete vehicles[i];

}

return 0;

}

#include <iostream>

#include <string>

using namespace std;

// Abstract base class

class Vehicle {

};

// Derived class: Car

class Car : public Vehicle{

};

// Derived class: Bike

class Bike : public Vehicle{

};

Question 2: Linked Lists (25 points total)

I asked ChatGPT to generate a question that would test your knowledge of linked lists in C++. Here is the question that it came up with , together with the solution that it generated:

You are tasked with implementing a singly linked list in C++. This linked list should support some basic operations and one advanced operation.

#include <iostream>

using namespace std;

struct Node {

int data; // TODO 1:

Node\* next; // TODO 2:

// TODO 3:

Node(int value) : data(value), next(nullptr) {}

};

class LinkedList {

private:

Node\* head; // TODO 4:

public:

// TODO 5:

LinkedList() : head(nullptr) {}

// TODO 6:

void insertAtHead(int value) {

Node\* newNode = new Node(value); // TODO 7:

newNode->next = head; //TODO 8:

head = newNode; //TODO 9:

}

// TODO 10:

void insertAtPosition(int value, int position) {

Node\* newNode = new Node(value);

// TODO 11:

if (position == 0) {

newNode->next = head;

head = newNode;

return;

}

// TODO 12:

Node\* temp = head;

for (int i = 0; i < position - 1 && temp != nullptr; i++) {

temp = temp->next;

}

// TODO 13:

if (temp == nullptr) {

cout << "Error: Position out of bounds!" << endl;

delete newNode;

return;

}

// TODO 14:

newNode->next = temp->next;

temp->next = newNode;

}

// TODO 15:

void display() {

Node\* temp = head;

while (temp != nullptr) {

cout << temp->data << " -> ";

temp = temp->next;

}

cout << "NULL" << endl;

}

// TODO 16:

void reverse() {

Node\* prev = nullptr;

Node\* current = head;

Node\* next = nullptr;

while (current != nullptr) {

next = current->next; // TODO 17:

current->next = prev; // TODO 18:

prev = current; // TODO 19:

current = next; // TODO 20:

}

head = prev;

}

// TODO 21:

~LinkedList() {

Node\* temp;

while (head != nullptr) {

temp = head;

head = head->next;

delete temp;

}

}

};

int main() {

LinkedList list;

// TODO 22:

list.insertAtHead(10);

list.insertAtHead(20);

list.insertAtHead(30);

// TODO 23:

cout << "Original list: ";

list.display();

// TODO 24:

list.insertAtPosition(25, 1);

cout << "After inserting 25 at position 1: ";

list.display();

// TODO 25:

list.reverse();

cout << "Reversed list: ";

list.display();

return 0;

}

Your task is to complete the comments in the above code (indicated in 25 places with TODO) so as to demonstrate that you understand how the code works.