**Name: Hamia Khalil**

**Reg No:l1s23bscs0070**

**“Lab 2”**

Task 1:

#include <iostream>

using namespace std;

class Shape {

public:

virtual void area() = 0;

};

class circle :public Shape {

private:

int radius;

public:

circle(int r = 0) {

radius = r;

}

void area() {

float Area;

Area = 3.14 \* radius \* radius;

cout << "Area of circle is: " << Area << endl;

}

};

class rectangle :public Shape {

private:

int length;

int width;

public:

rectangle(int l=0,int w=0){

length = l;

width = w;

}

void area() {

float Area;

Area = length \* width;

cout << "Area of rectangle is: " << Area << endl;

}

};

int main() {

circle c(5);

rectangle r(3, 2);

Shape\* s;

s = &c;

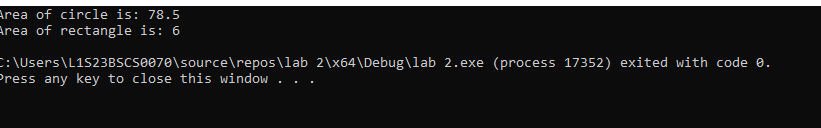
s->area();

s = &r;

s->area();

return 0;

}

Task 2:

#include <iostream>

using namespace std;

class Employee {

public:

virtual void calculateSalary() = 0;

};

class FullTimeEmployee :public Employee {

private:

int salary;

public:

FullTimeEmployee(int s = 0) {

salary = s;

}

void calculateSalary() {

cout << "Salary of FullTimeEmployee is: " << salary << endl;

}

};

class HalfTimeEmployee :public Employee {

private:

int working\_hours;

int hourly\_rate;

public:

HalfTimeEmployee(int h=0,int r=0){

working\_hours = h;

hourly\_rate = r;

}

void calculateSalary() {

int s;

s = 26\*working\_hours \* hourly\_rate;

cout << "calculateSalary of HalfTimeEmployee is: " <<s << endl;

}

};

int main() {

FullTimeEmployee full(80000);

Employee\* s;

s = &full;

s->calculateSalary();

int h, r;

cout << "enter hourly rate of half time employee: ";

cin >> r;

cout << "enter working hours of half time employee: ";

cin >> h;

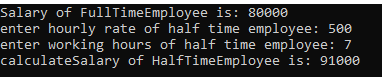
HalfTimeEmployee half (h, r);

s = &half;

s->calculateSalary();

return 0;

}



Task 3:

#include <iostream>

#include <string>

using namespace std;

// Abstract class

class Extra {

public:

virtual void display() const = 0; // Pure virtual function

virtual ~Extra() {}

};

// Book class

class Book : public Extra {

private:

string name;

string author;

int page\_no;

public:

Book(string n = "", string a = "", int p = 0)

: name(n), author(a), page\_no(p) {}

void display() const override {

cout << "Book: " << name << " by " << author << " with pages: " << page\_no << endl;

}

int getPages() const {

return page\_no;

}

string getName() const {

return name;

}

};

// Newspaper class

class Newspaper : public Extra {

private:

string name;

string date;

string edition;

public:

Newspaper(string n = "", string d = "", string e = "")

: name(n), date(d), edition(e) {}

void display() const override {

cout << "Newspaper: " << name << " on " << date << " Edition: " << edition << endl;

}

string getName() const { return name; }

string getEdition() const { return edition; }

};

// Library class

class Library {

private:

Book books[100];

Newspaper newspapers[100];

int bookCount;

int newspaperCount;

public:

Library() : bookCount(0), newspaperCount(0) {}

void addBook(const Book& book) {

if (bookCount < 100) {

books[bookCount++] = book;

}

else {

cout << "Library is full for books."<<endl;

}

}

void addNewspaper(const Newspaper& newspaper) {

if (newspaperCount < 100) {

newspapers[newspaperCount++] = newspaper;

}

else {

cout << "Library is full for newspapers." << endl;;

}

}

void displayCollection() const {

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

books[i].display();

}

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

newspapers[i].display();

}

}

void sortBooksByPages() {

for (int i = 0; i < bookCount - 1; ++i) {

for (int j = 0; j < bookCount - i - 1; ++j) {

if (books[j].getPages() > books[j + 1].getPages()) {

swap(books[j], books[j + 1]);

}

}

}

}

void sortNewspapersByEdition() {

for (int i = 0; i < newspaperCount - 1; ++i) {

for (int j = 0; j < newspaperCount - i - 1; ++j) {

if (newspapers[j].getEdition() > newspapers[j + 1].getEdition()) {

swap(newspapers[j], newspapers[j + 1]);

}

}

}

}

Book\* searchBookByTitle(const string& title) {

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

if (books[i].getName() == title) {

return &books[i];

}

}

return nullptr;

}

Newspaper\* searchNewspaperByName(const string& name) {

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

if (newspapers[i].getName() == name) {

return &newspapers[i];

}

}

return nullptr;

}

};

int main() {

// Create book objects

Book book1("The Catcher in the Rye", "J.D. Salinger", 277);

Book book2("To Kill a Mockingbird", "Harper Lee", 324);

// Create newspaper objects

Newspaper newspaper1("Washington Post", "2024-10-13", "Morning Edition");

Newspaper newspaper2("The Times", "2024-10-12", "Weekend Edition");

// Create a library object

Library library;

// Add books and newspapers to the library

library.addBook(book1);

library.addBook(book2);

library.addNewspaper(newspaper1);

library.addNewspaper(newspaper2);

// Display the entire collection

cout << "Before Sorting:\n";

library.displayCollection();

// Sort books by pages and newspapers by edition

library.sortBooksByPages();

library.sortNewspapersByEdition();

cout << "\nAfter Sorting:\n";

library.displayCollection();

// Search for a book by title

Book\* foundBook = library.searchBookByTitle("The Catcher in the Rye");

if (foundBook) {

cout << "\nFound Book:\n";

foundBook->display();

}

else {

cout << "\nBook not found.\n";

}

// Search for a newspaper by name

Newspaper\* foundNewspaper = library.searchNewspaperByName("The Times");

if (foundNewspaper) {

cout << "\nFound Newspaper:\n";

foundNewspaper->display();

}

else {

cout << "\nNewspaper not found.\n";

}

return 0;

}

