**Karnatak Law Society’s**

**GOGTE INSTITUTE OF TECHNOLOGY**

*Udyambag Belagavi -590008*

*Karnataka, India*



A Course Project Report on

**TITLE**

Submitted for the requirements of 4th semester B.E. in CSE

for **“OBJECT ORIENTED PROGRAMING USING C++ (18ACS28)”**

**Submitted by**

**NAME USN**

**1) Paritosh kumar 2GI20CS082**

**2) Parishkar Singh 2GI20CS081**

**3) Prashant 2GI20CS092**

**4) Parth Pawar 2GI20CS085**

**Under the guidance of**

**Mrs. Manjula Ramannavar**

**Asst. Prof., Department of CSE**

**Academic Year 2020-2021 (Even semester)**

**Table of contents**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Item** | **Page** |
| **1** | **Program on class and object** | **5** |
| **2** | **Program illustrating the use of reference types in C++** | **9** |
| **3** | **Program on function overloading** | **12** |
| **4** | **Program on dynamic memory management in C++** | **15** |
| **5** | **Program on array of objects** | **20** |
| **6** | **Program on constructors and destructors** | **27** |
| **7** | **Program on operator overloading** | **30** |
| **8** | **Program on overloading i/o operators** | **35** |
| **9** | **Program on inheritance** | **42** |
| **10** | **Program on friend functions and virtual functions** | **48** |
| **11** | **Program on dynamic polymorphism** | **52** |
| **12** | **Program on file stream** | **55** |

***PROJECT 1:***

Q1.Develop a C++ program with a class called car having data members brand, model and

year. Write the appropriate member functions to set the values of the data members using

scope resolution operator and also display the details of the car using objects.

***Source code:***

class Car {private:

string brand, model;

int year;

public: void initialize(string brand, string model, int year) {

this->brand = brand;

this->model = model;

this->year = year;}

void dispDetails() {cout<< "Brand of the car:" << brand <<endl;

cout<< "Model of the car:" << model <<endl;

cout<< "Year of production:" << year <<endl; }};

int main() { Car c1;Car c2;

c1.initialize("Honda", "Sedan", 2018);

c2.initialize("Ford", "Figo", 2015);

c1.dispDetails();

c2.dispDetails();

return 0;}

OUTPUT:

**Brand of the car:Honda**

**Model of the car:Sedan**

**Year of production:2018**

**Brand of the car:Ford**

**Model of the car:Figo**

**Year of production:2015**

***PROJECT 2:***

Q2.  Write a c function exchange( int & int & int &) that uses reference variable to cyclically exchange values in 3 variables.Write associated main to this function.

Source code:

#include <iostream>

**using** **namespace** std;

**void** cyclicExchange(**int** &a, **int** &b, **int** &c)

{ **int** temp;

temp = a;

a = b;

b = c;

c = temp;

}

**int** main()

{ **int** a=10, b=20, c=30;

cout<< "Before exchange:" << endl;

cout<< "a=" << a << " b=" << b << " c=" << c << endl;

cyclicExchange(a, b, c);

cout<< "After exchange:" << endl;

cout<< "a=" << a << " b=" << b << " c=" << c << endl;

**return** 0;}

OUTPUT:

**Before exchange:**

**a=10 b=20 c=30**

**After exchange:**

**a=20 b=30 c=10**

***PROJECT 3:***

Q3. Write the following overloaded functions:

    i. strcopy(s1, s2) that copies all the characters of s2 to s1

    ii. strcopy(s1, s2, n) that copies the first n characters of s2 to s1

   Write the associated main to exercise these functions.

**Source code:**

#include <iostream>

#include<cstring>

**using** **namespace** std;

**void** strcopy(**char** s1[], **char** s2[]) {

**int** i=strlen(s1), j;

**for**(j=0; j<strlen(s2); j++, i++)

  s1[i]=s2[j]; s1[i]='\0';}

**void** strcopy(**char** s1[], **char** s2[], **int** n) { **int** i=strlen(s1), j;

**for**(j=0; j<n; j++, i++)

  s1[i]=s2[j];

    s1[i]='\0';}

**int** main() {

**char** s1[] = "Hello", s2[]="World";

    strcopy(s1,s2);

    cout<< s1 <<endl;

    strcopy(s1,s2,3);

    cout<< s1 <<endl;

**return** 0;}

**Output:**

**HelloWorld**

**HelloWorld**

***PROJECT 4:***

Q(4)..Develop a C++ program to read records of n items as per the following structure:

struct Item{

            int itemCode;

            string name;

            float price;

};

Create the array of structures using dynamic memory allocation. Write functions

a.    to read the records and

b.    display item having highest price.

**Source code:**

struct Item {

int itemCode;

string name;

float price;

};

void readItems(Item a[], int n) {

cout<< "Enter " << n << " items:" <<endl;

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

cout<< "Item Code:";

cin>> a[i].itemCode;

cout<< "Name of the item:";

cin>> a[i].name;

cout<< "Price:";

cin>> a[i].price;

}

}

void findHighest(Item a[], int n) {

int highest = a[0].price, itemNum;

for(int i=1; i<n; i++)

if(a[i].price> highest) {

highest = a[i].price;

itemNum = i;

}

cout<< "Item with highest price:" <<endl;

cout<< "Item Code:" << a[itemNum].itemCode<<endl;

cout<< "Item Name:" << a[itemNum].name <<endl;

cout<< "Price:" << a[itemNum].price<<endl;

}

int main() {

int n;

cout<< "How many items?";

cin>> n;

Item \*a = new Item[n];

readItems(a, n);

findHighest(a, n);

return 0;

}

**Output:**

**How many items?-**3

**Enter 3 items:**

**Item Code:**101

**Name of the item:**pen

**Price:**10

**Item Code:**102

**Name of the item:**paper

**Price:**2

**Item Code:**103

**Name of the item:**Calculator

**Price:**80

**Item with highest price:**

**Item Code:103**

**Item Name:Calculator**

**Price:80**

***PROJECT 5:***

Q(5)..Create a class called Course that has following data members:

courseName, CourseCode, Faculty, numStudRegistered

Write member function to read data members and a friend function that returns the course

details where maximum registrations are done.

Create an array of Course class and exercise the above functions.

Source code:

**#include<iostream>**

**#include<string>**

**using namespace std;**

**class course { private: string name,code,faculty;**

**int nor;**

**public: void setdata() {cin.ignore();**

**cout<<"enter course name"<<endl;**

**getline(cin,name);**

**cout<<"enter course code"<<endl;**

**getline(cin,code);**

**cout<<"enter course faculty"<<endl;**

**getline(cin,faculty);**

**cout<<"enter no of student registered for course"<<endl;**

**cin>>nor; }**

**void display()**

**{cout<<"the details of max student registered course:"<<endl;**

**cout<<"course name:-"<<name<<endl;**

**cout<<"course code:-"<<code<<endl;**

**cout<<"course faculty:-"<<faculty<<endl;**

**cout<<"students registered are:-"<<nor<<endl;}**

**int maximum(course ar[],int n)**

**{int max,x; max=ar[0].nor;**

**for(int i=0;i<n;i++) {**

**if(ar[i].nor>max)**

**x=i; }**

**return x; }**

**int main()**

**{int n,x;**

**cout << "enter number of courses"<<endl;**

**cin>>n;**

**course a[n];**

**for(int i=0;i<n;i++)**

**{**

**a[i].setdata(); }**

**x=maximum(a,n);**

**a[x].display();**

**return 0;**

**}**

**Output:**

**enter number of courses:**3

**enter course name:**cse

**enter course code:**101

**enter course faculty:**mike

**enter no of student registered for course:**25

**enter course name:**phy

**enter course code:**102

**enter course faculty:**geeta

**enter no of student registered for course:**34

**enter course name:**chem

**enter course code:**103

**enter course faculty:**Manoj

**enter no of student registered for course**

28

**the details of max student registered course:**

**course name: chem**

**course code: 103**

**course faculty: Manoj**

**students registered are: 28**

***PROJECT 6:***

Q6. Create a class called Employee with id, name, designation, salary as the data members.

Write the following methods/constructors with the main function.

a. Default constructor to initialize the values for manager (employee)

b. Parameterized constructor to initialize the values of employees with other

designations.

c. A display\_detail function to display details of the employee.

d. Usage of destructor.

**Source code :**

class Employee { private: int id;

string name, desg;

float salary;

public: Employee() {

id = 0;name = "";

desg= ""; salary = 0.0f;}

Employee(int id, string name, string desg, float salary) { this->id = id; this->name = name;

this->desg = desg; this->salary = salary;}

void dispDetails() {

cout<< "Emplyee ID:" << id <<endl;

cout<< "Emplyee Name:" << name <<endl;

cout<< "Designation:" <<desg<<endl;

cout<< "Salary:" << salary <<endl;}

~Employee() {

cout<< "Employee object destroyed" <<endl;}};

int main() {Employee e(1010, "SRM", "Professor",34650.50f);

e.dispDetails();

return 0;}

**OUTPUT:**

**Emplyee ID:1010**

**Emplyee Name:SRM**

**Designation:Professor**

**Salary:34650.5**

**Employee object destroyed**

***PROJECT 7:***

Q7. Design an invoice class of items with array of price as the data member. Overload ‘&amp;’

operator to add the item price to the array and similarly overload ‘ ! ’ operator to remove an

item’s price from the array. Exercise the same using the main function.

**Source code :**

class Items {

private: int \*prices, i, n;

public: Items(int len) {

n = len; prices = new int[n];

i = 0;}

void operator &(int p) {if(i==n)

cout<< "Cannot add " << p << " as array is full" <<endl;

else {

prices[i++] = p;

cout<< "Added " << p << " into array" <<endl;}}

void operator !() {

if(i==0)

cout<< "Cannot remove as array is empty" <<endl;

else { i--;

cout<< prices[i] << " removed from array" <<endl;}}

void dispPrices() {

cout<< "Prices are..." <<endl;

for(int j=0; j<i; j++)

cout << prices[j];

cout<<endl;

}

};

int main() {

Items p(5);

p&10;

p&20;

p&30;

p&40;

p&50;

p&60;

p.dispPrices();

!p;

!p;

!p;

!p;

!p;

!p;

p.dispPrices();

return 0;

}

**OUTPUT:**

**Added 10 into array**

**Added 20 into array**

**Added 30 into array**

**Added 40 into array**

**Added 50 into array**

**Cannot add 60 as array is full**

**Prices are...**

**10 20 30 40 50**

**50 removed from array**

**40 removed from array**

**30 removed from array**

**20 removed from array**

**10 removed from array**

**Cannot remove as array is empty**

***PROJECT 8:***

Q8.. Create a class player (cricket) with player\_name , date of birth and ranking as data

members. Read the array of values for the data members mentioned. Display the player

details in Ascending order of ranking by overloading ‘&gt;&gt;’ operator and in descending order

of date of birth by overloading ‘&lt;&lt;’ operator. Exercise the same using the main function.

**Source code :**

class Player {

Private:string pName;

int ranking, n, yearOfBirth;

public: Player() {}

Player(int n) { this->n = n;}

friend ostream&operator <<(ostream&out, const Player p) {out<<p.pName<< " " <<p.yearOfBirth<< " " <<p.ranking<<endl;

return out;}

friend void sortOnYearOfBirth(Player p[], int n);

friend void swapMe(Player &p, Player &q);

friend void sortOnRanking(Player p[], int n);

friend istream&operator >>(istream&in, Player &p) {

cout<< "Enter player name:";

in>>p.pName; cout<< "Enter year of birth:";

in>>p.yearOfBirth; cout<< "Enter ranking:";

in>>p.ranking;return in;}};

void swapMe(Player &p1, Player &p2) {

Player temp;temp.pName = p1.pName;

temp.ranking = p1.ranking;

temp.yearOfBirth = p1.yearOfBirth;

p1.pName = p2.pName; p1.ranking = p2.ranking;

p1.yearOfBirth = p2.yearOfBirth;

p2.pName = temp.pName;

p2.ranking = temp.ranking;

p2.yearOfBirth = temp.yearOfBirth;}

void sortOnYearOfBirth(Player p[], int n) {

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

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

if(p[j].yearOfBirth< p[j+1].yearOfBirth)

swapMe(p[j], p[j+1]);}

void sortOnRanking(Player p[], int n) {

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

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

if(p[j].ranking> p[j+1].ranking)

swapMe(p[j], p[j+1]);}

int main() {

int n;

cout<< "How many players? ";

cin>> n;

Player \*p = new Player[n];

cout<< "Enter details of " << n << " players..."<<endl;

for(int i=0; i<n; i++)

cin>> p[i];

sortOnYearOfBirth(p, n);

cout<< "Players sorted on year of birth are..." <<endl;

for(int i=0; i<n; i++)

cout<< p[i];

sortOnRanking(p, n);

cout<< "Players sorted on ranking are..." <<endl;

for(int i=0; i<n; i++)

cout<< p[i];

return 0;

}

**OUTPUT:**

**How many players?** 4

**Enter details of 4 players...**

**Enter player name:**Sachin

**Enter year of birth:**1979

**Enter ranking:**2

**Enter player name:**Mitali

**Enter year of birth:**1985

**Enter ranking:**4

**Enter player name:**Dhoni

**Enter year of birth:**1983

**Enter ranking:**7

**Enter player name:**Mandhana

**Enter year of birth:**1990

**Enter ranking:**12

**Players sorted on year of birth are...**

**Mandhana 1990 12**

**Mitali 1985 4**

**Dhoni 1983 7**

**Sachin 1979 2**

**Players sorted on ranking are...**

**Sachin 1979 2**

**Mitali 1985 4**

**Dhoni 1983 7**

**Mandhana 1990 12**

***PROJECT 9:***

Q9..The class Cylinder inherits a superclass Circle. It further defines a variable

called height, a method computeVolume() and its own constructors.

Implement the hierarchy as shown below:

Circle

radius:double

Circle()

Circle(radius:double)

computeArea():double

Cylinder

height:double

Cylinder()

Cylinder(height:double)

Cylinder(height:double, radius:double)

computeVolume():double

**SOURCE CODE**

class Circle {

protected:

float radius, area;

public:

Circle() { radius = 0.0;}

Circle(float radius) {

this->radius = radius;}

void computArea() {

area = 3.142 \* radius \* radius;

cout << "Area of the circle is " << area << endl;}};

class Cylinder : public Circle {

private:

float height, volume;

public:

Cylinder() {

radius = 0.0; height = 0.0; }

Cylinder(float height) {

radius = 0.0;

this->height = height;

}

Cylinder(float radius, float height) : Circle(radius) {

this->height = height;}

void computeArea() {

area = 2 \* 3.142 \* radius \* height + 2 \* 3.142 \* radius \* radius;

cout << "Area of cylinder is " << area << endl;

}

void computeVolume() {

volume = 3.142 \* radius \* radius \* height;

cout << "Volume of cylinder is " << volume << endl;

}};

int main() {

Circle cir(4.5);

cir.computArea();

Cylinder cyl(4.5, 3.4);

cyl.computeArea();

cyl.computeVolume();

return 0;}

**OUTPUT:**

**Area of the circle is 63.6255**

**Area of cylinder is 223.396**

**Volume of cylinder is 216.327**

***PROJECT 10:***

Q(10) ..Implement the following inheritance hierarchy to demonstrate pure virtual functions.

Shape

x, y

computeArea()

Circle Triangle Rectangle

computeArea() computeArea() computeArea()

**Source code:**

class Shape {

protected:

float x, y, area;

public:

Shape(float x,float y=0) {

this->x = x;

this->y = y;}

virtual void computeArea()=0;};

class Circle : public Shape {

public:

Circle(float x) : Shape(x) {}

void computeArea() {

area = 3.142 \* x \* x;

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

}

};

class Rectangle : public Shape {

public:

Rectangle(float x, float y) : Shape(x, y) {}

void computeArea() {

area = x \* y;

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

}

};

class Triangle : public Shape {

public:

Triangle(float x, float y) : Shape(x, y) {}

void computeArea() {

area = 0.5 \* x\* y;

cout << "Area of triangle is " << area << endl;

}

};

int main() {

Circle c(4.5);

Rectangle r(4, 5);

Triangle t(3, 5.4);

c.computeArea();

r.computeArea();

t.computeArea();

return 0;}

OUTPUT:

**Area of circle is 63.6255**

**Area of rectangle is 20**

**Area of triangle is 8.1**

***PROJECT 11:***

Q.(11) Write a C++ program to create one superclass Animal and three subclasses, Herbivores,

Carnivores and Omnivores. Subclasses inherits from the superclass Animal and override its

eat() method. Demonstrate dynamic polymorphism to print data of Herbivores, Carnivores

and Omnivores by upcasting objects of the three derived types in main.

**Source code:**

#include <iostream>

**using** **namespace** std;

**class** Animal { **public**:

**virtual** **void** eat()

{ cout << "Animals Eat\n"; }};

**class** Herbivores : **public** Animal {

**public**:**void** eat() {

    cout << "Herbivores Eat Plants\n";}};

**class** Carnivores : **public** Animal {

**public**: **void** eat() {

cout << "Carnivores Eat Meat\n"; }};

**class** Omnivores : **public** Animal {

**public**: **void** eat() {

cout <<"Omnivores Eat Plants and Meat";}};

**int** main() {Animal a; Herbivores h;

Carnivores c;nOmnivores o;

Animal \*p; p = &a; p->eat();

p = &h; p->eat();

p = &c; p->eat();

p = &o;p->eat(); **return** 0;}

**OUTPUT:**

**Animals Eat**

**Herbivores Eat Plants**

**Carnivores Eat Meat**

**Omnivores Eat Plants and Meat**

**Program ended with exit code: 0**

***PROJECT 12:***

Q(12)..Write a C++ program to encrypt the contents of a file using substitution cipher. In a

Substitution cipher, any character of plain text from the given fixed set of characters is

substituted by some other character from the same set depending on a key. For example, with

a shift of 1, A would be replaced by B, B would become C, and so on. Accept the key from

the user to perform encryption.

***Source code:***

#include <iostream>

using namespace std;

string encrypt(string text, int s)

{ string result = "";

for (int i=0;i<text.length();i++)

{ if (isupper(text[i]))

result += char(int(text[i]+s-65)%26 +65);

else result += char(int(text[i]+s-97)%26 +97); }

return result;}

int main()

{ string text; int s;

cout<<"enter your text you want to encrypt: ";

cin>>text; cout<<"enter the no of shift: ";

cin>>s; cout << "Text : " << text;

cout << "\nShift: " << s;

cout << "\nCipher: " << encrypt(text, s);

return 0;}

**OUTPUT:**

**enter your text you want to encrypt:** computer

**enter the no of shift:** 2

**Text : computer**

**Shift: 2**

**Cipher: eqorwvgt**

**enter your text you want to encrypt:** goodday

**enter the no of shift:** 3

**Text : goodday**

**Shift: 3**

**Cipher:jrrggdb Program ended with exit code:0**