

OOP

Assignment#03



Name: Talib Husain

Roll# 21F-9070

**Task-1**

#include<iostream>

using namespace std;

#include<string>

static int c = 0;

void combinations(string a, int l, int r);

int main() {

string str;

cout << "Enter a String: ";

getline(cin, str);

int size = str.length()-1;

combinations(str,0,size);

cout << "Total Combnations: " << c << endl;

}

void combinations(string a, int l, int r)

{

if (l == r) {

cout << a << endl;

c = c + 1;

}

else

{

for (int i = l; i <= r; i++)

{

char temp = a[l];

a[l] = a[i];

a[i] = temp;

combinations(a, l + 1, r);

temp = a[l];

a[l] = a[i];

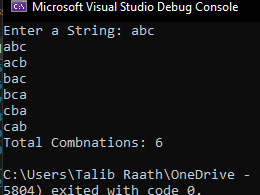
a[i] = temp;

}

}

}

**Output**



**Task-2 (A)**

#include<iostream>

#include<string>

using namespace std;

class Students {

string gender;

int long CNIC;

string name;

float GPA;

public:

Students() {

this->name = "";

this->CNIC = 0;

this->GPA = 0;

this->gender = "NIL";

}

Students(string name, string gender, int long cnic, float gpa) {

this->name = name;

this->gender = gender;

this->GPA = gpa;

this->CNIC = cnic;

}

string getname() {

return this->name;

}

float getGPA() {

return this->GPA;

}

int long getCNIC() {

return this->CNIC;

}

string getgender() {

return this->gender;

}

void setname(string name) {

this->name = name;

}

void setGPA(float gpa) {

this->GPA = gpa;

}

void setgender(string gender) {

this->gender = gender;

}

void setCNIC(int long cnic) {

this->CNIC = cnic;

}

void Dislay() {

cout << "Name: " << this->name << endl;

cout << "Gender: " << this->gender << endl;

cout << "CNIC: " << this->CNIC << endl;

cout << "GPA: " << this->GPA << endl;

}

void inputData() {

cout << "Enter a Name: ";

getline(cin, this->name);

cout << "Enter a gender: ";

getline(cin, this->gender);

cout << "Enter a CNIC: ";

cin>> this->CNIC;

cout << "Enter a GPA: ";

cin >> this->GPA;

}

};

class Section {

Students stdnt[40];

string sec\_name;

string teacher\_name;

static int count;

public:

Section() {

this->sec\_name = "";

this->teacher\_name = "";

count = 0;

}

void setSectionName(string sec\_name) {

this->sec\_name = sec\_name;

}

void setTeacherName(string teacher\_name) {

this->teacher\_name = teacher\_name;

}

string getSectionName() {

return this->sec\_name;

}

string getTeacherName() {

return this->teacher\_name;

}

void addStudent() {

this->stdnt[count].inputData();

count++;

}

void editSection() {

cout << "Enter New Section Name: ";

getline(cin, this->sec\_name);

cout << "Enter New Teacher Name: ";

getline(cin, this->teacher\_name);

}

void update(string name) {

int s = search\_student(name);

if (s > -1)

stdnt[s].inputData();

else

cout << "This student does't exist\n";

}

void printlist() {

cout << this->sec\_name << endl;

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

cout << i << "-";

stdnt[i].Dislay() ;

}

}

int search\_student(string name) {

int c = -1;

for (int i = 0; i < 40; i++) {

if (stdnt[i].getname() == name){

c = i;

break;

}

}

return c;

}

};

int Section::count = 0;

int main() {

int NoOfSec, option = 0;

Section\* sections;

cout << "Enter a Number of Sections: ";

cin >> NoOfSec;

sections = new Section[NoOfSec];

string secname;

string temp;

int count = 0,c=0;

do

{

cout << "1-Edit Section Attributes\n2-Add a Student\n3-Update Student of section\n4-Print List of Section\n";

cout << "5-Print List of Students\n6-Exit\nChoose any Option: ";

cin >> option;

cin.ignore();

if (option == 1) {

cout << "Enter a Section Name: ";

getline(cin, secname);

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

if (sections[i].getSectionName() == secname)

sections[i].editSection();

}

}

else if (option == 2) {

if (count == 0){

cout << "Enter a Section Name: ";

getline(cin, temp);

sections[c].setSectionName(temp);

cout << "Enter a Teacher Name: ";

getline(cin, temp);

sections[c].setTeacherName(temp);

c++;

}

sections[c].addStudent();

count++;

cout << "count = " << count << endl;

if (count == 3) {

count = 0;

}

}

else if (option == 3) {

cout << "Enter a Student Name: ";

getline(cin, temp);

sections->update(temp);

}

else if (option == 4) {

cout << "Section Name Teacher Name\n";

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

cout << sections[i].getSectionName() <<" ";

cout<< sections[i].getTeacherName() << endl;

}

}

else if (option == 5) {

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

sections[i].printlist();

}

}

else {

exit(1);

}

cout << "\n";

system("pause");

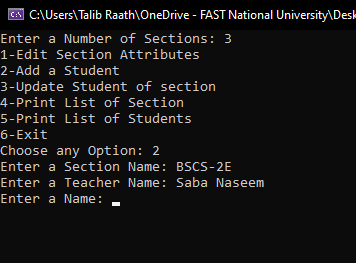
system("cls");

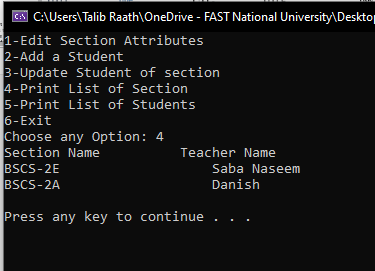
} while (option!=6);

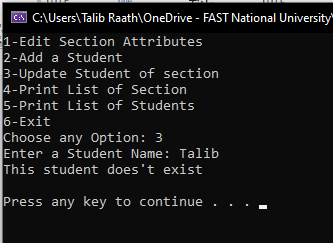
return 0;

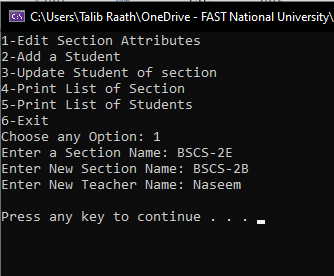
}

**Output**

****

****

****

****

**Task-2 (B)**

#include<iostream>

using namespace std;

class Matrix {

int row, col;

int\*\* ptr;

public:

Matrix() {

ptr = nullptr;

row = 0;

col = 0;

}

Matrix(Matrix& m) {

this->row = m.row;

this->col = m.col;

ptr = new int\* [this->row];

for (int i = 0; i < this->col; i++) {

\*(this->ptr + i) = new int[this->col];

}

for (int i = 0; i < m.row; i++) {

for (int j = 0; j < m.col; j++) {

(this->ptr[i][j] )= m.ptr[i][j];

}

}

}

void transpose();

void Build(int row, int col);

void input();

~Matrix() {

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

delete[]ptr[i];

}

delete[]ptr;

}

};

void Matrix::Build(int row, int col) {

this->row = row;

this->col = col;

ptr = new int\*[row];

for (int i = 0; i < this->col; i++) {

\*(this->ptr + i) = new int[col];

}

}

void Matrix::input() {

cout << "Enter Elements" << endl;

for (int i = 0; i < this->row; i++)

{

for (int j = 0; j < this->col; j++) {

cin >> ptr[i][j];

}

}

}

void Matrix::transpose() {

for (int i = 0; i< this->col; i++) {

for (int j= 0; j < this->row; j++) {

cout << this->ptr[j][i] << " ";

}

cout << endl;

}

cout << endl;

}

int main() {

Matrix m1;

Matrix m2;

int rows, cols;

cout << "Enter Rows: ";

cin >> rows;

cout << "Enter Col: ";

cin >> cols;

m1.Build(rows, cols);

m2.Build(rows, cols);

m1.input();

m2.input();

cout << "Transpose of Matrix-A\n";

m1.transpose();

cout << "Transpose of Matrix-B\n";

m2.transpose();

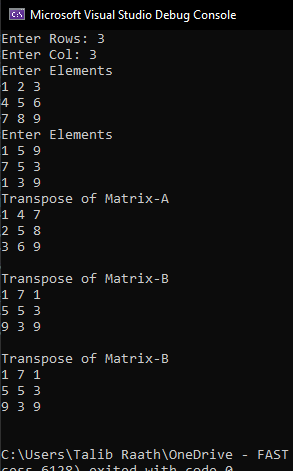
Matrix m3 = m2;

cout << "Transpose of Matrix-B\n";

m3.transpose();

}

**Output**

****

**Task-3**

#include<iostream>

#include<iomanip>

using namespace std;

struct Timer {

int hours;

int minutes;

int seconds;

};

class Testing {

int test\_id;

string question;

Timer time;

public:

void setTestID(int ID) {

this->test\_id = ID;

}

void setquestion(string question) {

this->question = question;

}

void setTimer(int hours, int minutes, int seconds) {

this->time.hours = hours;

this->time.minutes = minutes;

this->time.seconds = seconds;

}

int getId() {

return this->test\_id;

}

string getquestion() {

return this->question;

}

Timer gettime() {

return time;

}

void Display() {

cout << this->test\_id << setw(5)<<' ';

cout << this->question << setw(5)<<' ';

cout << this->time.hours << ":" << this->time.minutes << ":" << this->time.seconds<<endl;

}

};

int main() {

int no;

cout << "Enter Number of Question: ";

cin >> no;

Testing\* tests;

tests = new Testing[no];

for (int i = 0; i < no; i++) {

tests[i].setTestID(i+1);

tests[i].setquestion("This is question no");

tests[i].setTimer(2, 3, 4);

}

for (int i = 0; i < no; i++) {

cout << tests[i].getId() << " ";

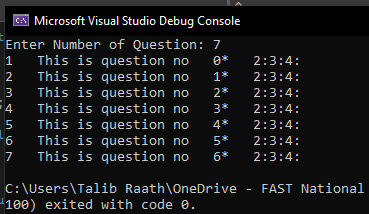
cout << tests->getquestion() <<" "<< i << "\*" << " ";

cout << tests[i].gettime().hours << ':' << tests[i].gettime().minutes << ':' << tests[i].gettime().seconds << ':' << endl;

}

}

**Output**

****

**Task-4**

#include<iostream>

using namespace std;

class Example {

private:

int data;

static int count;

public:

Example(int y = 10) : data(y) {

}

int getIncrementedData() const

{

//We can only access the the member function but cannot modify the value in const function

return data;

}

static int getCount() {

//We can't access non-static data menber in static member function

// cout << "Data is " << data << endl;

return count;

}

};

int Example::count = 0;

int main() {

Example e(4);

cout<<Example::getCount()<<endl;

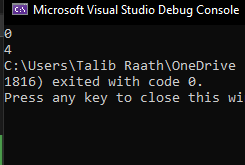
cout<<e.getIncrementedData();

}

**Exaplanation**

1. static data members should be initialized.
2. **Error in const member fun: We can only access the member function but cannot modify the value in the const function. Pre increment in data member is trying to modify the value that’s why it is giving an error.**
3. **We can't access non-static data members in a static member function. That’s why it is giving an error. We can resolve it by making data member static. Or only accessing the static data member.**

**Now the program is running without any error**

****

**Task-5**

#include<iostream>

using namespace std;

class Account {

double balance;

public:

Account(double balance) {

//checking the balance is valid or not

if (balance >= 0)

this->balance = balance;

else {

this->balance = 0;

cout << "Balance was invalid\n";

}

}

void Credit(double credit) {

cout << "Attempting to Debit $" << credit << endl;

if (credit >= 0)

this->balance += credit;

else

cout << "Invalid amount\n";

}

bool Debit(double debit) {

bool x = 1;

cout << "Attempting to Debit $" << debit << endl;

if(debit<=this->balance && debit>=0)

this->balance -= debit;

else {

cout << "Debit amount exceeded account balance\n";

x = 0;

}

return x;

}

double getbalance() {

return this->balance;

}

};

//CLASS FOR SAVING ACCOUNT

class SavingAccount : public Account {

double interstrate;

public:

SavingAccount(double balance, double interrate) : Account(balance) {

this->interstrate = (interrate/100);

}

double CalculateInterest() {

cout << "Calculation Interest\n";

return this->interstrate \* this->getbalance();

}

};

//CLASS FOR CHECKING ACCOUNT

class CheckingAccount: public Account {

double feecharged;

public:

CheckingAccount(double balance, double fee): Account(balance) {

this->feecharged = fee;

}

void Credit(double credit) {

//CALL THE CREDIT FUNCTION OF CLASS ACCOUNT To Add a credit

Account::Credit(credit);

}

void Debit(double debit) {

if(Account::Debit(debit));

}

};

int main() {

SavingAccount account1(100006, 5);

SavingAccount account2(500000, 4);

CheckingAccount account3(152789, 10);

cout << "Balance in account1: " << account1.getbalance() << endl;

cout << "Balance in account2: " << account2.getbalance() << endl;

cout << "Balance in account3: " << account3.getbalance() << endl<<endl;

account1.Credit(1000);

account1.Debit(5000);

account2.Credit(1000);

account2.Debit(-10000000);

account3.Credit(1000);

account3.Debit(5000);

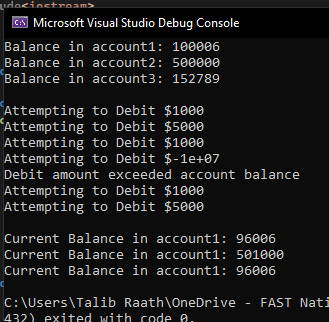
cout<<"\nCurrent Balance in account1: "<<account1.getbalance() << endl;

cout << "Current Balance in account1: " << account2.getbalance() << endl;

cout << "Current Balance in account1: " << account1.getbalance() << endl;

}

**Output**

****

**Task-6**

#include<iostream>

using namespace std;

class Package {

protected:

string name\_sender,address\_sender,city\_sender,state\_sender;

string name\_receiver, address\_receiver, city\_receiver, state\_receiver;

int ZIP\_code\_sender;

int ZIP\_code\_receiver;

double weight,cost;

public:

Package(){

ZIP\_code\_receiver = 0;

ZIP\_code\_sender = 0;

weight = 0;

cost = 0;

}

void set\_sender(string name, string address, string city, string state, int zip\_sender) {

this->address\_sender = address;

this->city\_sender = city;

this->state\_sender = state;

this->name\_sender = name;

this->ZIP\_code\_sender = zip\_sender;

}

void set\_receiver(string name, string address, string city, string state, int zip\_receiver) {

this->address\_receiver = address;

this->city\_receiver = city;

this->state\_receiver = state;

this->name\_receiver = name;

this->ZIP\_code\_receiver = zip\_receiver;

}

void setcost(double cost) {

this->cost = cost;

}

void setweight(double weight) {

this->weight = weight;

}

double CalculateCost() {

return this->weight \* this->cost;

}

void display\_sender() {

cout << "Sender:\n";

cout << this->name\_sender<<endl;

cout << this->address\_sender << endl;

cout << this -> city\_sender << endl;

}

void display\_receiver() {

cout << "\nReceipient:\n";

cout << this->name\_receiver << endl;

cout << this->address\_receiver << endl;

cout << this->city\_receiver << endl;

}

};

class TwoDayPackage : public Package {

double flat\_fee;

public:

TwoDayPackage() {

flat\_fee = 0;

}

void setFlatFee(double fee) {

this->flat\_fee = fee;

}

double CalculateCost() {

return Package::CalculateCost() + this->flat\_fee;

}

};

class OvernnightPackage : public Package {

double additional\_fee;

public:

OvernnightPackage() {

additional\_fee = 0;

}

void setAddtionalFee(double Addfee) {

this->additional\_fee = Addfee;

}

double CalculateCost() {

this->cost += this->additional\_fee;

return Package::CalculateCost();

}

};

int main() {

Package package1;

TwoDayPackage package2;

OvernnightPackage package3;

package1.setcost(17);

package1.setweight(300);

package1.set\_sender("Talib", "365 B-Block Johar Town", "Lahore", "Punjab", 68932);

package1.set\_receiver("Ahmad", "Fast-Nu", "CFD Campus", "Faisalabad", 56532);

cout << "Package-1" << endl;

package1.display\_sender();

package1.display\_receiver();

cout << "Cost: " << package1.CalculateCost() << endl << endl;

package2.setcost(17);

package2.setweight(300);

package2.set\_sender("Talib", "365 B-Block Johar Town", "Lahore", "Punjab", 68932);

package2.set\_receiver("Ahmad", "Fast-Nu", "CFD Campus", "Faisalabad", 56532);

cout << "Package-2" << endl;

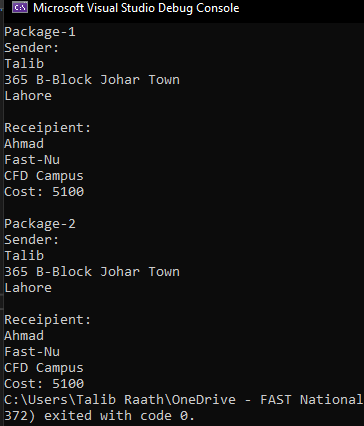
package2.display\_sender();

package2.display\_receiver();

cout << "Cost: " << package2.CalculateCost();

}

**Output**

****