OOPS MINI PROJECT: MARKS MANAGEMENT

NAMES: 1. DANIEL CHANDY (20CE18)

2. DION MORAES (20CE20)

CODE:

#include <iostream>

#include <string>

#include <vector>

#include <iomanip>

using namespace std;

class Student {

string name;

int rollno;

public:

Student (string n, int r){

name = n;

rollno = r;

}

void display () const {

cout << "Name : " << name << " roll Number : " << rollno << endl << endl;

}

void setName (string n) {

name = n;

}

void setRollNo (int r) {

rollno = r;

}

string getName () const {

return name;

}

int getRollNumber () const {

return rollno;

}

};

class Engineering {

protected:

float mathMarks;

float chemistryMarks;

float physicsMarks;

public:

Engineering (float a, float b , float c) : mathMarks {a} , chemistryMarks {b} , physicsMarks {c}

{

}

void display () const {

cout << "Mathematics Marks : " << mathMarks << " Chemistry Marks : " <<

chemistryMarks << " Physics Marks : " << physicsMarks << endl;

}

};

class ComputerEngineering : public Student , public Engineering { static int noCompEngStu;

float computerProgrammingMarks;

float computerOrganisationMarks;

float logicDesignMarks;

float percentage;

public:

ComputerEngineering (string n, int r,float m , float c , float p, float cp, float co , float ld):

Student (n,r) , Engineering(m,c,p) {

computerProgrammingMarks = cp;

computerOrganisationMarks = co;

logicDesignMarks = ld;

noCompEngStu++;

}

~ComputerEngineering(){

noCompEngStu--;

}

void display () const {

Student::display();

Engineering::display();

cout << "Computer Programming Marks : " << computerProgrammingMarks ;

cout << " Computer Organisation Marks : " << computerOrganisationMarks;

cout << " Logic Design Marks : " << logicDesignMarks << endl;

}

static int getNumberofCompStu () {

return noCompEngStu;

}

void calculatePercentage () {

float var = (mathMarks + chemistryMarks + physicsMarks + logicDesignMarks + computerOrganisationMarks + computerProgrammingMarks)/600.0;

percentage = var \* 100.0;

}

float getPercentage () const {

return percentage;

}

};

int ComputerEngineering::noCompEngStu = 0;

class mechanicalEngineering : public Student , public Engineering { static int noMechStu;

float thermodynamicsMarks;

float fluidMechanicsMarks;

float solidMechanicsMarks;

float percentage;

public:

mechanicalEngineering (string n, int r,float m , float c , float p, float thr, float fm , float sm):

Student (n,r) , Engineering(m,c,p) {

thermodynamicsMarks = thr;

fluidMechanicsMarks = fm;

solidMechanicsMarks = sm;

noMechStu++;

}

~mechanicalEngineering(){

noMechStu--;

}

void display () const {

Student::display();

Engineering::display();

cout << "Thermodynamic Marks : " << thermodynamicsMarks ;

cout << " Fluid Mechanics Marks : " << fluidMechanicsMarks ;

cout << " Solid Mechanics Marks : " << solidMechanicsMarks << endl;

}

static int getnoMechStu() {

return noMechStu;

}

void calculatePercentage () {

float var = (mathMarks + chemistryMarks + physicsMarks + thermodynamicsMarks + fluidMechanicsMarks + solidMechanicsMarks)/600.0;

percentage = var \* 100.0;

}

float getPercentage () const {

return percentage;

}

};

int mechanicalEngineering::noMechStu = 0;

class electricalEngineering : public Student , public Engineering { static int noElecStu;

float analysisOfCircuitsMarks;

float digitalElectronicsMarks;

float percentage;

public:

electricalEngineering (string n, int r,float m , float c , float p, float ancc, float dge ): Student (n,r) , Engineering(m,c,p) {

analysisOfCircuitsMarks = ancc;

digitalElectronicsMarks = dge;

noElecStu++;

}

~electricalEngineering(){

noElecStu--;

}

void display () const {

Student::display();

Engineering::display();

cout << "Analysis of circuit Marks : " << analysisOfCircuitsMarks ;

cout << " Digital Electronics Marks : " << digitalElectronicsMarks << endl;

}

static int getNoElecStu () {

return noElecStu;

}

void calculatePercentage () {

float var = (mathMarks + chemistryMarks + physicsMarks + analysisOfCircuitsMarks + digitalElectronicsMarks)/500.0;

percentage = var \* 100.0;

}

float getPercentage () const {

return percentage;

}

};

int electricalEngineering::noElecStu = 0;

void displayComEnghelper (vector <ComputerEngineering> c){ cout << setprecision(2) << fixed;

cout << setw(50) <<"Computer Engineering Department\n"; for (ComputerEngineering &comp : c) {

comp.calculatePercentage();

cout << setw(10) << left << "Name : " << setw(10) << comp.getName() << setw(15) << "RollNo : " << setw(10) << comp.getRollNumber() << setw(15) << " Percentage : " <<setw(10) <<

comp.getPercentage() << endl;

cout << endl;

}

}

void displayMechEnghelper (vector <mechanicalEngineering> m){ cout << setw(50) << right <<"Mechanical Engineering Department\n"; for (mechanicalEngineering mech : m) {

mech.calculatePercentage();

cout << setw(10) <<left << "Name : " <<setw(10) << mech.getName() <<setw(15) << "Roll No : " << setw(10) <<mech.getRollNumber() <<setw(15)<< "Percentage : " << setw(10)<< mech.getPercentage() << endl;

cout << endl;

}

}

void displayElecEnghelper (vector <electricalEngineering> e){

cout << setw(50) << right <<"Electrical Engineering Department\n"; for (electricalEngineering elct : e) {

elct.calculatePercentage();

cout << setw(10) << left <<"Name : " <<setw(10) << elct.getName() <<setw(15) << "Roll No:" <<setw(10) <<elct.getRollNumber() <<setw(15)<< " Percentage : " <<setw(10) <<elct.getPercentage() << endl;

cout << endl;

}

}

void compDept() {

vector <ComputerEngineering> compdept; int noOFStudents = 0;

cout<<"\nEnter no. of Computer Students: "; cin>>noOFStudents;

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

{

string name; int rollNo = 0;

float maths = 0.00, chem = 0.00 , phy = 0.00, ld= 0.00, cp = 0.00, co = 0.00; cout<<"\nEnter Name : ";

cin>>name;

cout<<"Enter Roll. No. : "; cin>>rollNo;

cout<<"Enter Marks out of 100"<<endl;

cout<<"Maths : ";

cin>>maths;

cout<<"Chemistry : ";

cin>>chem;

cout<<"Physics : ";

cin>>phy;

cout<<"Comp. Prog. : ";

cin>>cp;

cout<<"Comp. Org. : ";

cin>>co;

cout<<"Logic Design : ";

cin>>ld;

ComputerEngineering c(name,rollNo,maths,chem,phy,cp,co,ld); compdept.push\_back(c);

}

displayComEnghelper(compdept);

cout << endl << endl << endl;

}

void mechDept(){

vector <mechanicalEngineering> mechdept;

int noOFStudents = 0;

cout<<"\nEnter no. of Mechanical Students: ";

cin>>noOFStudents;

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

{

string name;

int rollNo = 0;

float maths = 0.00, chem = 0.00 , phy = 0.00, thr = 0.00, fm = 0.00, sm = 0.00;

cout<<"\nEnter Name : ";

cin>>name;

cout<<"Enter Roll. No. : ";

cin>>rollNo;

cout<<"Enter Marks out of 100"<<endl;

cout<<"Maths : ";

cin>>maths;

cout<<"Chemistry : ";

cin>>chem;

cout<<"Physics : ";

cin>>phy;

cout<<"Thermo Dynamics: ";

cin>>thr;

cout<<"Fluid Mechanics : ";

cin>>fm;

cout<<"Solid Mechanics : ";

cin>>sm;

mechanicalEngineering m (name,rollNo,maths,chem,phy,thr,fm,sm); mechdept.push\_back(m);

}

displayMechEnghelper(mechdept);

cout << endl << endl << endl;

}

void elecDept () {

vector <electricalEngineering> elc;

int noOFStudents = 0;

cout<<"\nEnter no. of Electronic Students: ";

cin>>noOFStudents;

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

{

string name;

int rollNo = 0;

float maths = 0.00, chem = 0.00 , phy = 0.00, ancc = 0.00, dge = 0.00;

cout<<"\nEnter Name : ";

cin>>name;

cout<<"Enter Roll. No. : ";

cin>>rollNo;

cout<<"Enter Marks out of 100"<<endl;

cout<<"Maths : ";

cin>>maths;

cout<<"Chemistry : ";

cin>>chem;

cout<<"Physics : ";

cin>>phy;

cout<<"Analysis Of Circuits: ";

cin>>ancc;

cout<<"Digital Electronics : ";

cin>>dge;

electricalEngineering e (name,rollNo,maths,chem,phy,ancc,dge); elc.push\_back(e);

}

displayElecEnghelper(elc);

cout << endl << endl << endl;

}

int main () {

char c;

do {

cout<<"\n@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\n";

cout <<"C - Computer Department \n";

cout << "M - Mechanical Department\n";

cout << "E - Electrical Department\n";

cout << "Q - Quit \n\n";

cout<<"\n@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@\n";

cin >> c;

if ( c == 'C' || c == 'c'){

compDept();

}

else if (c == 'M' || c == 'm'){

mechDept();

}

else if (c == 'E' || c == 'e'){

elecDept();

}

else if (c == 'Q' || c == 'q'){

exit(1);

}

else {

cout << "\nInvalid choice\n\n";

}

}while (c != 'Q');

return 0;

}

OUTPUTS:Text, table

Description automatically generated

Text, letter

Description automatically generated Text, letter

Description automatically generated Text

Description automatically generated

Block Diagram:

**class Student**

name: string

roll no: int

-----------------------------------------------

display()

setname()

setrollno()

getname()

getrollno()

Student()

**class mechengineering**

mosprogmarks: float

engthermomarks: float

emsmarks: float

percentage: float

-------------------------------------------------

display()

getnosh()

calculate-percentage()

get-percentage()

mechengineering()

**class compengineering**

compprogmarks: float

comarks: float

ldmarks: float

percentage: float

-------------------------------------------------

display()

getnosh()

calculate-percentage()

get-percentage()

compengineering()

**class elecengineering**

casamarks: float

edcmarks: float

dssmarks: float

percentage: float

-------------------------------------------------

display()

getnosh()

calculate-percentage()

get-percentage()

elecengineering()

**class Engineering**

mathMarks: float

chemistryMarks: float

physicsMarks: float

---------------------------------------------

display()

Engineering()