**Ministerul Educației al Republicii Moldova**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică și Microelectronică**

**Departamentul Ingineria Software și Automatică**

**Raport**

Lucrarea de laborator nr.5

Disciplina Programarea orietntată pe obiecte:

Tema: Moştenirea multiplă

Varianta 5

**Efectuat**: st.gr. TI-207 Bunescu Gabriel

**Verificat lect.univ.Lisnic Inga**.

Chișinău 2021

**Scopul lucrării:**

* Studierea regulilor de determinare a moştenirii multiple;
* Studierea avantajelor şi neajunsurilor moştenirii multiple;
* Probleme legate de utilizarea moştenirii multiple;
* Studierea rezolvării problemelor;

**Sarcina:**

Pentru toate variantele este necesar de creat două programe, care ar ilustra ambele exemple date mai sus de moştenire multiplă.

**Mersul lucrării:**

Varianta 5

1. Să se creeze, o ierarhie de moştenire: iahta, motocicleta – motocicleta de apa.
2. Să se creeze, o ierarhie de moştenire: transport – iahta, motocicletă – motocicleta de apa.

**Listingul programului:**

**a)**

#include <iostream>

#include <string.h>

#include <stdio.h>

#include <cstring>

using namespace std;

class iahta{

public: //declararea variabilelor ca fiinde publice, pentru a pute accesa din alta parte a progam.

char \*motor;

float hp;

char \*denumire;

float masa;

iahta():motor(),hp(),denumire(),masa(){}; //CONSTR. IMPLICID iahta(char \*denumire,int \*hp,char \*motor,int masa){ // CONSTR. EXPLICIT motor = new char; motor = new char[strlen(motor)+1];

iahta (char \*denumire,int \*hp,char \*motor,int masa){ motor = new char; motor = new char[strlen(motor)+1];

strcpy(motor,motor);

strcpy(motor,motor); //alocarea de memorie p/u un char hp = h

denumire = new char[strlen(denumire)+1]; //alocare p/u denumire

strcpy(denumire,denumire); //copie din denum in denumire masa=masa;

};

iahta(iahta &obj){ //CONSTR. DE COPIERE

motor = new char; strcpy(motor,obj.motor); hp=obj.hp;

denumire = new char[strlen(denumire)+1]; strcpy(denumire,obj.denumire);

masa=obj.masa;

};

~iahta(){ // DESTUCTOR

delete motor; motor=NULL; //sterge pointerul mot;

hp=0;

delete denumire; denumire=NULL;

masa=0; //masa i se atribuie (zero)

};

};

istream & operator>>(istream &in, iahta &obj){ char buf[100];

cout<<"Denumire: "; in>>buf;

obj.denumire = new char [strlen(buf)+1];

strcpy(obj.denumire,buf);

cout<<"Cai puteri: "; in>>obj.hp;

cout<<"Motor: "; obj.motor = new char;

in>>obj.motor;

cout<<"Masa: "; in>>obj.masa;

return in;

};

class motocicleta: public iahta{

char \*model;

public:

motocicleta(){model=NULL; };

motocicleta(char \*den,int hp,char \*motor,int masa,char \*model){

iahta::denumire = new char[strlen(den)+1];

strcpy(iahta::denumire,den);

iahta::hp = hp;

iahta::motor = new char;

strcpy(iahta::motor,motor);

iahta::masa = masa;

model = new char[strlen(model)+1];

strcpy(model,model);

};

friend istream & operator>>(istream &in, motocicleta &obj);

friend ostream & operator<<(ostream &out,motocicleta &obj);

};

istream & operator>>(istream &in, motocicleta & obj){

char buf[50];

cout<<"Denumirea: "; in>>buf;

obj.denumire = new char[strlen(buf)+1];

strcpy(obj.denumire,buf);

cout<<"hp c/p: "; in>>obj.hp;

cout<<"Motorul: "; in>>buf;

obj.motor = new char[strlen(buf)+1];

strcpy(obj.motor,buf);

cout<<"Greutatea (kg): "; in>>obj.masa;

cout<<"Modelul:"; in>>buf;

obj.model = new char[strlen(buf)+1];

strcpy(obj.model,buf);

return in; };

ostream & operator<<(ostream &out, motocicleta &obj){

//\*\*\*\*\*\*\*\*\*SUPRAINCARCAREA OPERATOR DE IESIRE COUT << ostream & operator<<(ostream &out, motocicleta &obj){

out<<"Denumire: "; out<<obj.denumire<<endl;

out<<"hp: "; out<<obj.hp<<endl;

out<<"Motorul: "; out<<obj.motor<<endl;

out<<"Greutatea: "; out<<obj.masa<<endl;

out<<"Modelul: "; out<<obj.model<<endl;

return out;

};

class motocicleta\_de\_apa: public iahta{

float viteza;

public:

motocicleta\_de\_apa(){viteza=0;};

motocicleta\_de\_apa(char \*den,int hp,char \*motor,int masa, int viteza){ iahta::denumire = new char[strlen(den)+1];

strcpy(iahta::denumire,den);

iahta::hp = hp;

iahta::motor = new char;

strcpy(iahta::motor,motor);

iahta::masa = masa;

viteza = viteza;

};

friend istream & operator>>(istream &, motocicleta\_de\_apa &);

friend ostream & operator<<(ostream &, motocicleta\_de\_apa &);

};

istream & operator>>(istream &in, motocicleta\_de\_apa &obj){ char buf[50];

cout<<"Denumirea: "; in>>buf;

obj.denumire = new char[strlen(buf)+1];

strcpy(obj.denumire,buf);

cout<<"hp: "; in>>obj.hp;

cout<<"Motorul: "; in>>buf;

obj.motor = new char;

strcpy(obj.motor,buf);

cout<<"Greutatea (kg): "; in>>obj.masa;

cout<<"Viteza max pe apa:"; in>>obj.viteza;

return in;

};

ostream & operator<<(ostream &out,motocicleta\_de\_apa &obj){

out<<"Denumirea: "; out<<obj.denumire<<endl;

out<<"hp: "; out<<obj.hp<<endl;

out<<"Motor: "; out<<obj.motor<<endl;

out<<"Greutatea: "; out<<obj.masa<<endl;

out<<"Viteza max pe apa:"; out<<obj.viteza<<endl;

return out;

};

int main(void){

char selection;

cout<<"1 "; cout<<"- Introdu Iahta "<<endl;

cout<<"2 "; cout<<"- Introdu motocicleta "<<endl;

cout<<"3 "; cout<<"- Introdu motocicleta de apa"<<endl;

cout<<"4 "; cout<<"- Iesire"<<endl;

cin >> selection;

switch (selection ) {

case '1' : {

iahta a;

cin>>a;

}

do {

cout << '\n' << "...\n";

} while (cin.get() != '\n');

case '2' : {

motocicleta b;

cin>>b;

cout<<b;

}

case '3' : {

motocicleta\_de\_apa c;

cin>>c;

cout<<c;

}

do {

cout << '\n' << "...\n";

} while (cin.get() != '\n');

case '4' : {

exit(1);

}

default : {

cout<<"\n Invalid selection";

cout<<"\n";

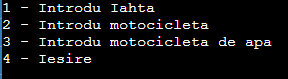
getchar();

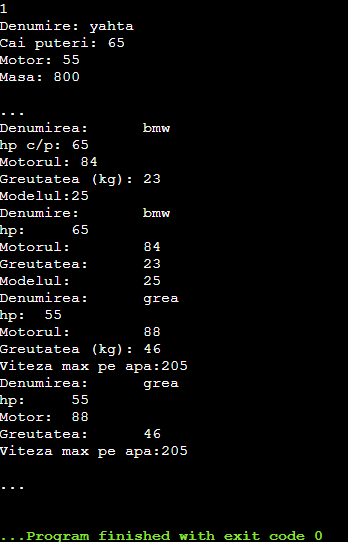
}

break;

}

}

****

****

b)

#include <iostream>

#include <string.h>

using namespace std;

class transport{

public:

char \*motor;

float hp;

char \*denumire;

float masa;

transport ():motor(NULL),hp(0),denumire(NULL),masa(0){};

transport (char \*denumire,int \*hp,char \*motor,int masa){ motor = new char; motor = new char[strlen(motor)+1];

strcpy(motor,motor);

hp = hp;

denumire = new char[strlen(denumire)+1];

strcpy(denumire,denumire);

masa=masa;

};

transport(transport &obj){

motor = new char; strcpy(motor,obj.motor);

hp=obj.hp;

denumire = new char[strlen(denumire)+1];

strcpy(denumire,obj.denumire);

masa=obj.masa;

};

~transport(){

delete motor; motor=NULL;

hp=0;

delete denumire; denumire=NULL;

masa=0;

};

friend ostream & operator<<(ostream &, transport &);

friend istream & operator>>(istream &, transport &);

};

ostream & operator<<(ostream &out, transport &obj){

cout<<"Denumirea: "<<obj.denumire<<endl;

cout<<"Cai puteri: "<<obj.hp<<endl;

cout<<"Motor: "<<obj.motor<<endl;

cout<<"Masa: "<<obj.masa<<endl;

return out;

};

istream & operator>>(istream &in, transport &obj){

char buf[100];

cout<<"Denumire: ";

in>>buf;

obj.denumire = new char [strlen(buf)+1];

strcpy(obj.denumire,buf);

cout<<"Cai puteri: ";

in>>obj.hp;

cout<<"Motor: ";

obj.motor = new char;

in>>obj.motor;

cout<<"Masa: ";

in>>obj.masa;

return in;

};

class iahta: public transport{

char \*col;

public:

iahta(){ col=NULL; };

iahta(char \*den, int hp, char \*motor, int masa, char \*col){

transport::denumire = new char[strlen(den)+1];

strcpy(transport::denumire,den);

transport::hp = hp;

transport::motor = new char;

strcpy(transport::motor,motor);

transport::masa=masa;

col = new char[strlen(col)+1];

strcpy(col,col);

};

friend istream & operator>>(istream &in, iahta &obj);

friend ostream & operator<<(ostream &out, iahta &obj);

};

istream & operator>>(istream &in, iahta & obj){

char buf[50];

cout<<"Denumirea: ";

in>>buf;

obj.denumire = new char[strlen(buf)+1];

strcpy(obj.denumire,buf);

cout<<"hp: ";

in>>obj.hp;

cout<<"motor: ";

in>>buf;

obj.motor = new char;

strcpy(obj.motor,buf);

cout<<"Greutatea: ";

in>>obj.masa;

cout<<"Culoarea : ";

in>>buf;

obj.col = new char[strlen(buf)+1];

strcpy(obj.col,buf);

return in;

};

ostream & operator<<(ostream &out, iahta &obj){

out<<"Denumirea: ";

out<<obj.denumire<<endl;

out<<"hp: ";

out<<obj.hp<<endl;

out<<"Motorul: ";

out<<obj.motor<<endl;

out<<"Greutate kg: ";

out<<obj.masa<<endl;

out<<"Culoarea: ";

out<<obj.col<<endl;

return out;

};

class motocicleta: public transport{

char \*model;

public:

motocicleta(){model=NULL; };

motocicleta(char \*den,int hp,char \*motor,int masa,char \*model){

transport::denumire = new char[strlen(den)+1];

strcpy(transport::denumire,den);

transport::hp = hp;

transport::motor = new char;

strcpy(transport::motor,motor);

transport::masa = masa;

model = new char[strlen(model)+1];

strcpy(model,model);

};

friend istream & operator>>(istream &in, motocicleta &obj);

friend ostream & operator<<(ostream &out,motocicleta &obj); };

istream & operator>>(istream &in, motocicleta &obj){ char buf[50];

cout<<"Denumirea: ";

in>>buf;

obj.denumire = new char[strlen(buf)+1];

strcpy(obj.denumire,buf);

cout<<"hp c/p: ";

in>>obj.hp;

cout<<"Motorul: ";

in>>buf;

obj.motor = new char[strlen(buf)+1];

strcpy(obj.motor,buf);

cout<<"Greutatea (kg): ";

in>>obj.masa;

cout<<"Modelul:";

in>>buf;

obj.model = new char[strlen(buf)+1];

strcpy(obj.model,buf);

return in;

}

ostream & operator<<(ostream &out, motocicleta &obj){

cout<<"Denumire: ";

out<<obj.denumire<<endl;

cout<<"hp: ";

out<<obj.hp<<endl;

cout<<"Motorul: ";

out<<obj.motor<<endl;

cout<<"Greutatea: ";

out<<obj.masa<<endl;

cout<<"Modelul: ";

out<<obj.model<<endl;

return out;

};

class motocicleta\_de\_apa: public transport{

float viteza;

public:

motocicleta\_de\_apa(){viteza=0;};

motocicleta\_de\_apa(char \*den,int hp,char \*motor,int masa, int viteza){

transport::denumire = new char[strlen(den)+1];

strcpy(transport::denumire,den);

transport::hp = hp;

transport::motor = new char;

strcpy(transport::motor,motor);

transport::masa = masa;

viteza = viteza;

};

friend istream & operator>>(istream &, motocicleta\_de\_apa &);

friend ostream & operator<<(ostream &, motocicleta\_de\_apa &); };

void getch();

istream & operator>>(istream &in, motocicleta\_de\_apa &obj){ char buf[50];

cout<<"Denumirea: "; in>>buf;

obj.denumire = new char[strlen(buf)+1];

strcpy(obj.denumire,buf);

cout<<"hp: "; in>>obj.hp;

cout<<"Motorul: "; in>>buf;

obj.motor = new char;

strcpy(obj.motor,buf);

cout<<"Greutatea (kg): ";

in>>obj.masa;

cout<<"Viteza max pe apa:";

in>>obj.viteza;

return in;

};

ostream & operator<<(ostream &out,motocicleta\_de\_apa &obj){

cout<<"Denumirea: ";

out<<obj.denumire<<endl;

cout<<"hp: ";

out<<obj.hp<<endl;

cout<<"Motor: ";

out<<obj.motor<<endl;

cout<<"Greutatea: ";

out<<obj.masa<<endl;

cout<<"Viteza max pe apa:";

out<<obj.viteza<<endl;

return out;

};

int main(void){

char selection;

cout<<"\n Menu";

cout<<"\n========";

cout<<"\n 1 - Introduceti denumirea transportului" <<endl;

cout<<"\n 2 - Iahta" <<endl;

cout <<"\n 3 - Motocicleta" <<endl;

cout <<"\n 4 - Motocicleta de apa" <<endl;

cout <<"\n 5 - Iesire" <<endl;

cout <<"\n Optiunea aleasa -----> " <<endl;

cin >> selection;

switch(selection) {

case '1' : {

transport a;

cout<<"\n Datele transportului" <<endl;

cin >> a;

cout << a;

}

do {

cout << '\n' << "...";

} while (cin.get() != '\n');

case '2' : {

iahta b;

cout<<"\n Datele iahtei" <<endl;

cin>>b;

cout<<b;

}

do {

cout << '\n' << "...\n";

} while (cin.get() != '\n');

case '3' : {

motocicleta c;

cout<<"\n Datele motocicletei" <<endl;

cin>>c;

cout<<c;

}

case '4' : {

motocicleta\_de\_apa d;

cout<<"\n Datele motocicletei de apa" <<endl;

cin>>d;

cout<<d;

}

do {

cout << '\n' << "...\n";

} while (cin.get() != '\n');

case '5' : {

exit(1);

}

default : {

cout<<"\n Invalid selection";

cout<<"\n";

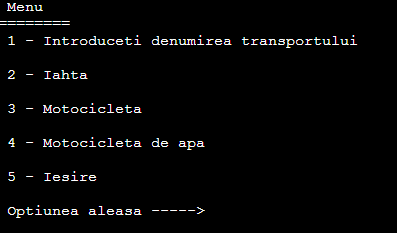
getchar();

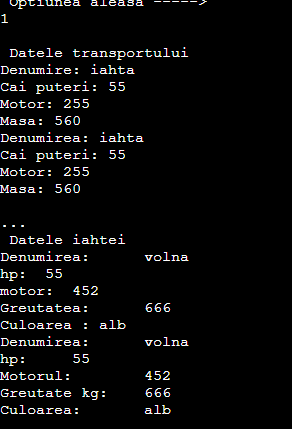
}

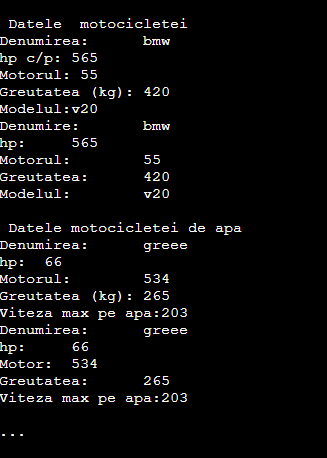
break;

}

}

****

****

****

**Concluzie:**

In urma efectuarii lucrarii date am determinat ca mediul de prgramare C++ este cu mult flexibil ca C, care ne permite sa manipulam cu informatia intr-un mod rapid si eficient fara a induce erori grosolane, si am aflat ca utilizarea claselor este cu mult mai comod si mai rapid in eleborarea unui program.

Am aflat ca moştenirea multiplă, reprezintă prin sine moştenirea de la două sau mai multe clase, si ca acest mecanism este foarte necesar. Dar, el nu se foloseşte în toate limbajele, dar este realizat în C++.