LUCRARE DE LABORATOR NR. 4

Tema: Moştenirea şi compoziţia

Scopul lucrării:

- studierea moştenirii, avantajele şi dezavantajele
- studierea compoziţiei
- studierea regulilor de definire a moştenirii şi compoziţiei
- studierea formelor de moştenire
- studierea iniţializatorilor
- principiul de substituţie
- moştenire şi compoziţie

Noţiuni de bază

Derivarea permite definirea într-un mod simplu, eficient şi flexibil a unor clase noi prin adăugarea unor funcţionalităţi claselor deja existente, fără să fie necesară reprogramarea sau recompilarea acestora. Clasele derivate exprimă relaţii ierarhice între conceptele pe care acestea le reprezintă şi asigură o interfaţă comună pentru mai multe clase diferite.

Moştenirea poate fi abordată din două puncte de vedere: al elaboratorului şi al utilizatorului clasei.

Din punctul de vedere al elaboratorului, moştenirea abordează comportarea şi proprietăţile clasei derivate ca extensie a proprietăţilor clasei de bază. Din punct de vedere al utilizatorului – moştenirea semnifică existenţa unui şir de clase parţial intershimbabile cu o interfaţă unică. (Inginerii TI şi inginerii mecanici sunt ambii, în primul rînd, ingineri).

Avantajele moştenirii:

- micşorarea volumului de cod şi utilizarea lui repetată,
- micşorarea numărului de erori şi a preţului de cost,
- interfaţă unică şi substituire,
- mărirea vitezei de elaborare.

Un neajuns al moştenirii ar fi o oarecare micşorare a vitezei de executare a codului. Totuşi eficienţa nu trebuie să contrazică avantajele, deoarece cheltuielile reale nu sunt esenţiale, ele fiind evitate parţial prin utilizarea funcţiilor.

Definirea şi utilizarea moştenirii

O clasă care asigură proprietăţi comune mai multor clase se defineşte ca o clasă de bază. O clasă derivată moşteneşte de la una sau mai multe clase de bază toate caracteristicile acestora, cărora le adaugă alte caracteristici noi, specifice ei.

În general, derivarea unei clase se specifică în felul următor:

```
class nume_derivata : specificator_acces nume_baza {
// corpul clasei
};
```

Specificatorul de acces poate fi unul din cuvintele-cheie: *public, private, protected*.

De exemplu:

```
class Animal{
    int NofLegs;
public:
    void Say(){ cout<<"!!!"; }
};
class Dog: public Animal{  // moştenire
...
};
void main(){
 Dog d;
 d.Say();
}
```

Din exemplu se vede că, funcţia şi variabila definite în clasa *Animal* sunt importate şi pot fi utilizate de obiectul clasei *Dog*. Clasa derivată poate să supradefinească comportarea definită în clasa de bază. (Struţul şi pinguinul nu zboară, deşi sunt păsări, dar ornitoringul depune ouă, deşi este mamifer).

Principiul de substituţie

Moştenirea mai presupune că, obiectul clasei derivate poate fi utilizat în locul obiectului clasei de bază:

```
void main(){
 Animal *ptr = new Dog;}
```

În exemplul dat se declară un pointer spre clasa de bază – deci se aşteaptă utilizarea obiectului clasei de bază, însă, pointerului i se atribuie obiectul clasei derivate.

Formele moştenirii

În C++ există cinci forme diferite de moştenire. În lucrarea de faţă vor fi analizate trei tipuri, numite moşteniri simple care au protecţia:

- public,
- private,
- protected,

Formele diferite de moştenire se utilizează pentru modificarea statutului de acces pentru elementele membru ale clasei.

Când o clasă este derivată dintr-o clasă de bază, clasa derivată moşteneşte toţi membrii clasei de bază (cu excepţia unora: constructori, destructor şi funcţia operator de asignare). Tipul de acces din clasa derivată la membrii clasei de bază este dat de specificatorul de acces. Dacă nu este indicat, specificatorul de acces este implicit *private*.

Când specificatorul de acces este *public*, toţi membrii de tip *public* ai clasei de bază devin membri de tip *public* ai clasei derivate; toţi membrii *protected* ai clasei de bază devin membri *protected* ai clasei derivate. Membrii *private* ai clasei de bază rămân *private* în clasa de bază şi nu sunt accesibili membrilor clasei derivate. Această restricţie de acces ar putea pare surprinzătoare, dar, dacă s-ar permite accesul dintr-o clasă derivată la membrii *private* ai clasei de bază, noţiunile de încapsulare şi ascundere a datelor nu ar mai avea nici o semnificaţie.

La utilizarea moştenirii de tip *private* a clasei de bază, toate elementele accesibile (*public* şi *protected*) din clasa de bază devin *private* în clasa derivată. Iar la o moştenire de tip *protected* – devin *protected*. Elementele cu acces *private* ale clasei de bază la orice formă de moştenire sunt inaccesibile în clasa derivată. Înafară de aceasta, formele de moştenire se deosebesc prin faptul că moştenirea de tip *public* crează un subtip de date, adică corespunde principiului de substituţie, iar moştenirea de tip *protected* şi *private* – nu.

Compoziţia

Compoziţia reprezintă încă un mecanism legat de POO, care desemnează relaţia dintre obiecte, pe cînd moştenirea este relaţia dintre clase.

Moştenirea realizează relaţia este *"is a". Cîinele* este *mamifer,* iar *mamiferul – animal*. Compoziţia realizează relaţia *conţine, are "has a". Automobilul* conţine *motor* şi *roţi.*

Definirea compoziţiei

De fapt compoziţia se aplică foarte larg, deoarece şi variabilele predefinite au tip şi se utilizează la definirea clasei. În cazul claselor utilizator, complexitatea constă în utilizarea constructorilor, definirea propriu-zisă fiind simplă:

```
class Car{
 Engine e;
};
```

Listele de iniţializare

Se ştie că variabilele clasei de bază protejate cu modificatorul *private* nu sunt accesibile, prin urmare, ele nu pot fi iniţializate în constructorul clasei derivate, mai mult ca atît, aceasta contrazice principiul de reutilizare a codului. Soluţia este una – apelul constructorului clasei de bază. Acest lucru are loc dar numai pentru constructorii impliciţi şi de copiere, generaţi de compilator. Restul constructorilor trebuie să se apeleze manual. Aceeaşi problemă apare şi în cazul compoziţiei, de asemenea poate fi utilizat numai constructorul implicit şi de copiere. Pentru soluţionarea acestor probleme sunt utilizate listele de iniţializare – care permit apelarea oricărui tip de constructor şi efectuează orice iniţializare.

```
class Engine{
 int power;
public:
 Engine(int p){power=p;}
};
class Transport{
...
public:
 Transport(char*);
};
class Car:public Transport{ // наследование
 Engine e;                  // композиция
public:
 Car():Transport("automobile"),e(10){}
};
```

Pentru apelul constructorului clasei de bază, după parantezele constructorului clasei derivate prin două puncte se scrie constructorul clasei de bază, fiind posibil transferul parametrilor din constructorul clasei derivate în constructorul clasei de bază. În cazul compoziţiei situaţia este asemănătoare dar se scrie numele variabilei şi nu a constructorului.

Ce alegem

Atît derivarea cît şi compoziţia reprezintă instrumente de reutilizarea a codului, apare o întrebare evidentă: cînd se utilizează moştenirea şi cînd – compoziţia. În acest sens, există multe recomandări diferite, însă cel mai simplu este să ne conducem de regula: se pune întrebarea: clasa nouă este un subtip (*Dog **is an** Animal*), dacă răspunsul este afirmativ, se utilizează moştenirea, în alt caz se pune întrebarea: clasa nouă reprezintă oare un container (*Car **has a** door*) – în acest caz se utilizează compoziţia.

Există însă situaţii cînd nici această strategie nu este suficientă. De exemplu, clasa *mulţime* poate fi creată în baza clasei *listă,* dar nu se ştie ce mecanism trebuie de folosit. Regulile în acest caz sunt mai complicate, se pune un set de întrebări:

- vor putea fi înlocuite obiectele clasei existente prin obiectele clasei noi?
- este necesar să se supradefinească careva funcâie virtuală?
- prelucrează oare tipul nou aceleaşi mesaje ca şi cel vechi?
- este oare clasa de bază abstractă?

Dacă răspunsurile sunt afirmative, se utilizează moştenirea.

Întrebări de control:

1. Prin ce se deosebesc formele de moştenire?
2. Ce reprezintă compoziţia ?
3. Care sunt avantajele moştenirii?
4. Cum lucrează constructorii la moştenire?
5. Ce reprezintă principiul de substituţie?
6. În ce cazuri se utilizează moştenirea şi în care se utilizează compoziţia?
7. Cum pot fi iniţializate elementele containerului?

Sarcina

Varianta 1

a) Să se creeze o ierarhie a claselor *joc – joc sportiv – volei*. Determinaţi constructorii, destructorul, operatorul de atribuire şi alte funcţii necesare.

b) Să se creeze clasa *roata,* care conţine rază. Determinaţi constructorii şi metodele de acces. Creaţi clasa *automobil,* care conţine roţi şi un câmp care reprezintă firma producătoare. Creaţi o clasă derivată *autocamion* care se deosebeşte prin tonaj. Determinaţi constructorii, destructorul şi alte funcţii necesare.

Varianta 2

a) Să se creeze clasa *student*, care are un nume, specialitate, anul de învăţământ şi balul mediu. Determinaţi funcţia de definire, schimbare a datelor şi de comparare. Pentru setarea câmpurilor textuale să se folosească operatorul new. Determinaţi constructorii, destructorul şi alte funcţii necesare. Creaţi clasa derivată *student-diplomant*, pentru care este definită tema de diplomă. De asemenea, este necesar de definit toate funcţiile necesare.

b) Să se creeze clasa *camera*, care conţine suprafaţă. Determinaţi constructorii şi metodele de acces. Creaţi clasa *apartament cu o odaie,* care conţine o odaie şi o bucătarie (suprafaţa ei), etajul (camera este în clasa apartament cu o odaie). Determinaţi constructorii, metodele de acces. Definiţi clasa derivată a apartamentelor cu o odaie cu adresă (un câmp adăugător - adresa). Determinaţi constructorii, destructorul şi fluxul de ieşire.

Varianta 3

a) Să se creeze clasa *mobilă,* care conţine informaţie despre preţ, stil şi domeniul de utilizare (oficiu, bucătărie şi altă mobilă). Pentru setarea câmpurilor textuale să se folosească memoria dinamică. Definiţi clasele derivate *masa* şi *scaun*. Definiţi constructorii, destructorul, operatorii de atribuire şi alte funcţii necesare.

b) Să se creeze clasa *garaj*, care conţine suprafaţa. Determinaţi constructorii şi metodele de acces. Creaţi clasa *casa*, care conţine odăi, o bucătarie (suprafaţa ei) şi garaj. Definiţi clasa derivată vilă (ca parametru adăugator – mărimea lotului de pamânt). Determinaţi constructorii, destructorul şi fluxul de ieşire.

Varianta 4

a) Să se creeze o ierarhie a claselor *om* şi *colaborator,* care ocupă un post anumit şi primeşte un salariu anumit. Să se supraîncarce operatorii pentru ieşiri şi intrări de obiecte, constructorul de copiere, operatorii de atribuire utilizînd funcţiile respective ale clasei de bază.

b) Să se creeze clasa *carte de joc*, care conţine grad şi culoare. Cartea poate fi întoarsă şi deschisă. Creaţi clasa – *butuc de cărţi*, care conţine cărţi de joc. Creaţi două clase derivate de la butuc de cărţi, în una cărţile pot fi scoase numai într-o ordine, iar în alta – aleator.

Varianta 5

a) Să se creeze clasa *lichid,* care conţine denumirea (pointer spre char), densitatea. Definiţi constructorii, destructorul, operatorii fluxului de intrare. Creaţi o clasă derivată – băuturi alcoolice, care conţine tărie. Determinaţi funcţiile de redenumire a densităţii şi a tăriei.

b) Să se creeze clasa *buton,* care conţine un text anumit. Definiţi constructorii şi metodele de acces. Creaţi clasa *fereastra,* care conţine un buton şi coordonatele ferestrei. Definiţi constructorii şi destructorul. Definiţi clasa derivată *fereastra cu buton şi cu un mesaj*. Definiţi constructorii, destructorul şi operatorii fluxului de ieşire.

Varianta 6

a) Să se creeze clasa *om,* care are numele (pointer spre char), vîrsta, greutatea. Determinaţi constructorii, destructorul şi operatorul de atribuire. Creaţi clasa derivata - *matur,* care are numărul paşaportului. Definiţi constructorii: impliciţi, cu parametri, destructorul, operatorii fluxului de ieşire. Definiţi funcţiile de resetare a vîrstei şi a numărului paşaportului.

b) Definiţi clasa *vaca* care este compusă din următoarele câmpuri: numărul de identificare – trebuie să fie garantat unic (pentru care se utilizează un contor static), cantitatea medie de lapte, vîrsta, numele şi rasa. Pentru setarea câmpurilor textuale utilizaţi operatorul new. Definiţi clasa cireadă, care constă dintr-un număr nelimitat de vaci. Definiţi metodele de introducere, eliminare, de determinare a cantităţii medii de lapte în cireadă şi cantitatea totală de lapte în cireadă şi alte funcţii necesare.

Varianta 7

a) Să se creeze o ierarhie a claselor *clădire –clădire admnistrativă – clădire de locuit*. Să se supraîncarce fluxul de ieşire şi fluxul de intrare, să se definească constructorul de copiere, operatorul de atribuire prin funcţiile corespunzătoare ale clasei de bază.

b) Să se creeze clasa *student,* care are nume, specialitate, anul de studiu şi balul mediu. Determinaţi funcţia de definire, schimbare a datelor şi comparare. Pentru setarea câmpurilor textuale se foloseşte operatorul new. Definiţi constructorii, destructorul şi alte funcţii necesare. Creaţi clasa *grupa,* care conţine studenţi (un număr nelimitat). Definiţi metodele de adăugare şi eliminare a studenţilor, de determinare a balului mediu pe grupă, constructorii, destructorul şi alte funcţii necesare.

Varianta 8

a) Să se creeze ierarhia de clase *bloc de studiu* - clasa de bază abstractă şi *instituţie preşcolară*, *instituţie şcolară medie* şi *instituţie de învăţămînt superior* – clase derivate. Să se supraîncarce fluxul de ieşire şi fluxul de intrare, să se definească constructorul de copiere, operatorul de atribuire prin funcţiile corespunzătoare ale clasei de bază.

b) Să se creeze clasa *motor,* care are o firmă producătoare, tip şi putere. Determinaţi funcţiile de setare, de schimbare a parametrilor motoarelor. Creaţi o ierarhie a claselor: corabie – clasa de bază şi vapor pentru pasageri – derivată. Corabia are un motor, capacitate de încărcare,

măsurător de apă, denumire, portul unde este înscris. Pentru setarea cîmpurilor textuale, utilizaţi operatorul new.

Varianta 9

a) Să se creeze o ierarhie a claselor *presă – ziar, revistă şi ziar electronic*. Determinaţi câmpurile: denumirea ziarului, tirajul, indexul de abonare, periodicitatea de publicare. Pentru setarea cîmpurilor textuale, utilizaţi operatorul new. Definiţi fluxul de ieşire şi fluxul de intrare, constructorul de copiere, operatorul de atribuire prin funcţia corespunzătoare a clasei de bază.

b) Să se creeze clasa *Procesor,* care conţine informaţia despre denumirea procesorului, frecvenţa lui, tehnologiile de producere utilizate şi mărimea memoriei. Determinaţi clasa *computer* , care este compusă dintr-un procesor şi alte componente. Pentru setarea cîmpurilor textuale, utilizaţi operatorul new. Definiţi constructorii, funcţia fluxului de ieşire şi alte funcţii necesare.

Varianta 10

a) Să se creeze o ierarhie a claselor *transport – transport aerian – elicopter*. Definiţi fluxul de ieşire şi fluxul de intrare, constructorul de copiere, operatorul de atribuire prin funcţia corespunzătoare a clasei de bază.

b) Definiţi clasa *element chimic*, care conţine informaţia despre denumirea elementului şi proprietăţile lui chimice. Determinaţi clasa *medicamente*, care conţine un număr diferit de elemente chimice şi în cantităţi diferite. Determinaţi constructorii, funcţiile fluxului de ieşire şi alte funcţii necesare.

Varianta 11

a) Să se creeze o ierarhie a claselor *senzor* - clasa de bază abstractă şi senzori de temperatură, de umiditate şi de măsurare a vitezei vântului. Pentru fiecare clasă să se definească unităţile sale de măsură şi metoda de citire a datelor despre starea mediului înconjurător. Să se supraîncarce fluxul de ieşire şi fluxul de intrare, constructorul de copiere, operatorul de atribuire prin funcţiile corespunzătoare ale clasei de bază.

b) Să se creeze clasa *dispozitiv de colectare a informaţiei* despre starea timpului care este compusă din senzori (conform sarcinei a). Pentru citirea valorilor să se creeze clasa *generatorul de valori* pentru fiecare senzor. Să se arate funcţionarea dispozitivului.

Varianta 12

a) Să se creeze o ierarhie a claselor *Figură de şah* - clasa abstractă, care conţine câmpul – culoare. Creaţi pentru toate figurile clase derivate , care conţin denumirea figurii şi coordonatele poziţiei pe tablă. Pentru setarea cîmpurilor textuale, utilizaţi operatorul new. Determinaţi fluxul de ieşire şi fluxul de intrare, constructorul de copiere, operatorul de atribuire prin funcţiile corespunzătoare ale clasei de bază.

b) Să se creeze clasa *figuri de şah,* care este compusă din setul de figuri din sarcina *a*, şi tabla de șah – tablou bidimensional 8 pe 8. Să se creeze posibilitatea de a elimina figurile de pe tablă. Definiţi constructorul care creează dinamic figuri şi le dă poziţii utilizînd notarea pentru şah (E2). Definiţi constructorul de copiere şi operatorul de atribuire.