**Descrieți pe scurt cum se comportă constructorii de copiere la moştenire.**

Constructorul implicit de copiere atât pentru bază cât și pentru derivată copiază bit cu bit. Cel derivat (implicit) apelează constructorul de copiere al bazei. Dacă constructorul bazei este suprascris, derivată îl va apela implicit. Dacă constructorul derivatei este suprascris, acesta apelează implicit constructorul de inițializare al bazei. Programatorul trebuie să definească în lista de inițializare a constructorului derivatei apelarea constructorului de copiere al bazei.

**Descrieți pe scurt operatorul typeid.**

Operatorul typeid face parte din biblioteca typeinfo și se folosește pentru a identifica tipul unui obiect (RTTI). El returnează un obiect de tipul type\_info care are un număr de metode publice dintre care cea mai folosită este name() care diferă de la compilator la compilator și conține numele clasei de care aparține obiectul. Tipul obiectului poate fi predefinit sau definit de programator, sintaxa de folosire este typeid(obiect) iar typeid(NULL) aruncă excepția bad\_typeid.

**Spuneți ce este obiectul implicit al unei metode şi descrieți pe scurt proprietătile pe care le cunoaşteți despre acesta.**

Obiectul implicit al unei metode se referă la instanța clasei de care aparține, reținută în pointer-ul constant “this”. Pointer-ul nu se poate modifica (astfel obiectul implicit nu poate fi modificat) și poate fi neprecizat în interioriul metodei (este recunoscut implicit). Metodele statice nu au obiect implicit deci nici pointer-ul this.

**Descrieți pe scurt funcțiile virtuale şi scopul în care sunt folosite.**

Funcțiile virtuale reprezintă un concept important POO, dinamizând codul, și sunt folosite pentru polimorfism la execuție și pentru a preveni object slicing-ul. Unei instanțe de clasă care conține funcții virtuale i se asociază un vtable care conține pointeri către corpul funcțiilor virtuale care va fi folosit la apelarea funcției. Acestea sunt declarate în clasa de bază sub forma de “virtual ret\_tip functie(param)” și sunt redefinite în clasele derivate fără prefixul “virtual”. Spre deosebire de supraîncărcarea funcțiilor, la funcțiile virtuale, tipul returnat trebuie să fie la fel (excepție tipuri derivate). Nu pot exista în exteriorul unei clase.

sau

Metodele virtuale reprezinta unul dintre cele 3 cazuri de polimorfism la executie. Sunt folosite in cadrul unei ierarhii de clase, fiind rescrise in clasele derivate, cu scopul de a putea decide, la momentul rularii, in functie de tipul obiectului apelant, varianta specializata a metodei. (in lipsa mecanismului de metoda virtuala, se va apela mereu varianta din clasa de bază, chiar dacă se face upcast deoarece se face object slicing). Varianta din clasa de baza e precedata de cuvantului cheie "virtual". Mecanismul acestora consta in faptul ca fiecare instanta a clasei retine în vtable adresa corpului functiei care se va executa la apelul metodei.

**Descrieți pe scurt funcțiile constante si dați un exemplu de utilizare.**

Funcțiile constante au sintaxa ret\_tip functie(param) const, și promit nemodificarea stării obiectului de care aparțin. În general, orice metodă ce nu modifică date ale obiectului de care aparțin ar trebui să fie const, exemplu: gettere, funcții de afișare.

**Descrieți pe scurt diferența dintre un pointer şi o referință.**

Un pointer este un obiect care reține o adresă și căruia i se atribuie spațiu în memorie pe când referința este un alias pentru o zonă de memorie deja existentă. O referință trebuie să fie inițializată la declarare pe când un pointer poate să nu fie.

**Descrieți trei metode de proiectare diferite prin care elementele unei clase se pot regăsi în dublu exemplar, sub diverse forme, în definiția altei clase.**

???

**Descrieți pe scurt moştenirea virtuală şi scopul în care este folosită.**

Moștenirea virtuală este folosită în cazul moștenirii multiple, atunci când dintr-o clasă B derivă două obiecte D1 și D2 din care derivă o clasă MM. Dacă moștenirea nu ar fi virtuală, obiectul MM ar avea în componența sa două instanțe ale clasei B. Prin moștenire virtuală ne asigurăm că lui B i se atribuie o singură zonă de memorie.

**Enumerați 3 metode de implementare a polimorfismului de compilare.**

Polimorfismul la compilare se realizează prin early binding și apare la supraîncărcarea constructorilor, supraîncarcarea funcțiilor și șabloanele.

**Descrieți pe scurt mecanismul de tratare a excepțiilor.**

Pentru a trata excepțiile folosim o structură try{...}catch(exceptie){...} care încearcă să execute o anumită porțiune de cod în interiorul try-ului iar la eventuala apariție a unor excepții (împărțirea la 0, cast-uri invalide, excepții generate de programator (prin folosirea throw)) se va ieși din blocul try și se va încerca potrivirea tipului de excepție cu ramurile blocului catch. După terminarea catch-ului nu se mai revine în interiorul blocului try iar dacă blocul try se termină fără excepții nu se va intra în blocul catch. Ramurile blocului catch nu fac conversie la tipurile de dată ale excepțiilor, dar fac upcast, astfel excepțiile derivate se pun înaintea bazelor.

sau

Prin tratarea exceptiilor se intelege stabilirea comportamentului programului in cazul unor situatii neprevazute aparute pe parcursul rularii (ex: alocarea unui vector cu numaru neagtiv de elemente, exceptii matematice - impartire la 0 etc.). Cuvintele cheie in mecanismul de tratare a exceptiilor sunt "try" (in blocul de try se incearca executarea unor instructiuni), "throw" (se arunca o exceptie), "catch" (se prinde exceptia si se trateaza). Pot aparea mai multe blocuri catch, distinctia fiind facuta de tipul parametrului (al valorii aruncate). De asemenea, exista si varianta catch(...), care poate prinde orice tip de eroare

**Descrieţi pe scurt ce reguli verifică supraîncărcarea operatorilor.**

Supraîncărcarea operatorilor se poate realiza atât ca metode membre (în care participantul stâng la operație este obiectul \*this) cât și ca funcții externe (în care participantul stâng se menționează ca parametru). Dacă nu este operator unar, ceilalți participanți la operație se precizează ca parametrii. Operatorii care nu pot fi suprascriși sunt **.**, **::**, **.\***, **?.** Operatorii care nu pot fi suprascriși ca funcții externe sunt =, (), [], ->. Funcțiile externe pot să aibă acces la date fie prin gettere și settere fie fiind definite ca friend în clasa asupra căreia operează. Dacă o funcție poate fi și membră și externă, se preferă a fi membră, dar există cazuri în care le preferăm externe, mai exact când dorim ca primul operand să nu fie obiectul implicit. Dacă vrem să avem de exemplu operația int + obiect, aceasta poate fi definit doar ca funcție externă. Nu se poate redefini precedența operatoriilor sau numărul de operanzi.

**Spuneţi dacă o variabilă constantă poate fi transmisă ca parametru al unei funcţii şi dacă da, în ce situaţii. Justificaţi.**

Da, dacă parametrul este la rândul său declarat constant sau dacă este un parametru prin valoare, întrucât variabila constantă va fi copiată.

**Spuneţi ce reprezintă o funcţie prietenă (friend) a unei clase.**

O funcție externă clasei care primește acces la datele private și protected a clasei în care se declară friend.

**Spuneţi pe scurt prin ce se caracterizează un câmp static al unei clase.**

Un câmp static este o variabilă împărțită de toate instanțele unei clase, pentru care se alocă în memorie o singură zonă pentru toate instanțele. Un câmp static poate fi apelat prin Clasa::variabilă sau ca o variabilă normală, din orice instanță obiect.variabilă. Câmpul static trebuie inițializat în afara clasei.

**Spuneţi ce reprezintă o funcţie virtuală şi în ce condiţii o funcţie virtuală defineşte o clasă abstractă.**

O funcție virtuală este o funcție care oferă polimorfism la execuție, adresa corpului ei fiind salvat în vtable și apelat la execuție. O funcție virtuală definește o clasă abstractă atunci când aceasta este pură (virtual ret\_tip functie(param)=0). Clasa abstractă nu poate fi instanțiată și este folosită doar pentru interfață.

**Descrieţi pe scurt constructorul de copiere şi situaţiile în care acesta este folosit.**

Constructorul de copiere este un constructor care creează un obiect nou pornind de la un obiect deja existent. Constructorul de copiere implicit copiază bit cu bit informațiile obiectului copiat și apelează constructorii de copiere ai claselor pe care le moștenește și pe care le are agregate. Dacă este redefinit de programator, el nu va mai copia implicit bit cu bit. Constructorii de copiere se redefinesc atunci când avem date alocate dinamic.

**Descrieţi pe scurt cum puteţi prelua o dată prin incluziune şi a doua oară prin moştenire o clasă numar într-o clasă lista care descrie liste nevide de dimensiune variabilă de elemente de tip numar.**

În cazul incluziunii folosim gettere pentru a prelua datele din clasa agregată, iar în cazul moștenirii folosim instanța clasei de număr moștenită de clasa listă pentru a prelua direct valorile. Nu putem prelua date private din clasa număr decât dacă facem clasa listă friend sau prin gettere sau transformarea datelor private în protected.

**Descrieţi pe scurt metoda de identificare a tipului în timpul rulării (RTTI).**

Run-Time Type Information se folosește pentru polimorfism la execuție, atunci când există situații în care tipul unui obiect nu se poate cunoaște de la compilare. El se realizează prin operatorul typeid(obiect) cu care putem identifica și compara tipurile de date atât predefinite cât și definite de programator.

**Spuneţi care este diferenţa dintre incluziunea de clase şi moştenirea de clase şi când se foloseşte fiecare metodă.**

Incluziunea este o relație de tipul “has a” în care în interiorul unui obiect există obiecte de tipul altei clase. Ea este folosită pentru a reutiliza cod, folosindu-ne de datele încapsulate în clasa agregată înloc să redefinim în noua clasa proprietățile de care avem nevoie din clasa agregată.

Moștenirea este o relație de tipul “is a” și constă într-o clasă derivată care moștenește proprietățile clasei de bază. Ea este folosită tot pentru reutilizare de cod și polimorfism la execuție.

**Spuneţi pe scurt prin ce se caracterizează o metodă statică a unei clase.**

O metodă statică nu este legată de o instanță și astfel nu are pointer-ul this. Totodată, ea poate folosi doar date membre statice și poate fi apelată fie prin Clasa::functie sau ca o metodă normală prin ob.metoda.

**Descrieţi pe scurt diferenţa dintre transferul prin valoare şi transferul prin referinţă al parametrilor în cazul apelului unei funcţii.**

Apelul prin referință nu apelează constructorul de copiere la intrarea în funcție și destructorul la părăsirea funcției, pe când apelul prin valoare o face. Un parametru transmis prin valoare este o variabilă locală și nu poate fi returnată prin referință. Apelul cu parametru referință neconst nu poate fi apelat cu valori const pe când cel prin valoare poate deoarece obiectul constant se copiază.

**Spuneţi care este diferenţa dintre clasa generică (template) şi clasa abstractă şi în ce situaţii se foloseşte fiecare dintre ele.**

Clasa generică template se folosește pentru generalizarea codului, în cadrul algoritmilor care realizează același lucru pentru mai multe tipuri de date. Clasa abstractă este o clasă folosită doar pentru interfața sa, neputând fi instanțiată, din ea moștenindu-se alte clase care folosesc clasa abstractă pentru polimorfism la execuție. O clasă generică poate fi instanțiată și specializată.

**Descrieţi pe scurt cum se pot defini funcţii de conversie între tipuri (clase).**

Pentru a face conversie între tipuri putem să ne folosim de upcast (realizat prin cast direct), downcast (realizat prin static\_cast, dynamic\_cast) și de reinterpret\_cast (care schimbă în totalitate tipul obiectului).

**Descrieţi pe scurt diferenţa dintre funcţiile care returnează valoare şi cele care returnează referinţă.**

Funcțiile care returnează referință nu pot să returneze variabile locale sau parametrii transmise prin valori deoarece vor fi distruse la ieșirea din funcție și zona referinței ar fi inaccesibilă, pe când cele care returnează prin valoare copiază obiectul returnat deci nu au această problemă.