Laboratorul 9

Clase Abstracte, how to ace your exam

# Takeaways



* Vom folosi cât mai simplu clasele abstracte cu așa-zisele „funcții virtuale pure”;
* Ce este pur la ele: faptul ca una dintre ele nu poate face nimic, deoarece nu are cod

# Exerciții

## 0. Clasele abstracte – exemplul cu formele de copt

În laboratorul 7 (de la 143), aveam o cerință simplu pe care este probabil s-o fi auzit și la seminar. Anume că avem mai multe tipuri de forme de copt (pătratice, dreptunghiulare, rotunde etc), și ni se cerea să aflăm aria totala și volumul prăjiturilor.

Am folosit moștenire (datele comune sunt înălțimea formei), dar și funcție virtuală pentru ariaBazei(), deoarece aria diferă în funcție de tipul formei.

CakeShape -> SquareShape

-> RectangleShape

-> CircleShape

In Laborator 9 -> Clase Abstracte -> Laborator7\_solved avem exercitiul din Laborator7, rezolvat cu ajutorul claselor abstracte.

**Am comentat cu // TODO Informațiile noi despre clase abstracte**

## Adăugăm moștenire multiplă claselor abstracte

Vom continua codul de la exercițiul 0

Ne dorim să formăm prăjituri care au baza rotundă și formă de cilindru de la bază până la înălțimea **height**, dupa care au formă de con de la înălțimea **height** până la înălțimea totală de **height\*2** cm.

Se cere:

* Creați o clasă ConeShape, care va fi forma de tort pentru varful prăjiturii mai speciale

!!! Ce are diferit clasă ConeShape diferit față de celelalte trei (care au forma de cub, paralelipiped, cilindru) ?

* Creăm prăjitura specială, prin moștenirea virtuală a claselor cu baza cerc, respectiv a formei ConeShape. Vom moșteni virtual, deoarece ne dorim să păstrăm o singură variabilă denumite **height** provenită din clasa de bază CakeShape.

Aceasta este moștenirea diamant. Adică

CakeShape -> ConeShape

-> CircleShape

Iar cele două clase Cone si Circle sunt ambele folosite în SpecialShape

**ConeShape** **---> \**

**|** **--->** **SpecialShape** (calculeaza si con, si cilindru)

**CircleShape** **---> /**

Moștenirea virtuală o veți scrie ca:

*class* SpecialShape : *public virtual* ConeShape, *public virtual* CircleShape {}

// ROLUL **virtual** este ca nu va duplica și variabila height, adică ConeShape::height și CircleShape::height vor fi aceiasi variabilă.

Exact după cum v-ați aștepta, când scrieți cod în clasa **SpecialShape** puteți apelă și **ConeShape::volume()**, și **CircleShape::volume()**, adică aveți acces la ambele clase.

## Tablouri în ulei pentru fiecare

**Bonus +10p**, deoarece lucram de la zero acest exercitiu.

Similar cu exercitiul 1, vom folosi functii virtuale pure.

### ***Takeaways despre metode și metode virtuale:***

**Cum ne dăm seama dacă niște date sunt atribute sau funcții?**

* **Dacă se calculeaza din alte atribute, avem de implementat o funcție.**

E.g. aria bazei nu poate să fie atribut, deoarece când modificăm o latură, ar trebui schimbată și ea. Ceea ce ar fi prea complex și fără rost.

*Menționez că există și ideea de caching. Putem avea și un atribut, și o funcție, dar adăugăm și reguli de recalculare a variabilei când se modifică atributele de care depinde. Pe scurt, prea multă bătaie de cap. Folosim mai ales când limbajul permite. Spre exemplu, Python permite să adăugăm simpla caching.*

**Când este o funcție virtuală?**

* **Când are efect diferit în funcție de derivata din care este apelată.**

E.g. ne putem gândi chiar la citirea/ scrierea virtuală

**Când este o funcție virtuală pură?**

1. **Când în clasă părinte nu putem calcula/ afișa ceea ce ne cere, deoarece nu avem toate informațiile. (pure = lacks any implementation)**

E.g. nu putem calcula salariul net, dacă nu știm rolul angajatului.

## **Cerința**

S-a deschis un magazin care automatizează crearea de tablouri în ulei. Folosește atât inteligență artificială modernă, precum [**DALL-E**](https://openai.com/blog/dall-e/)**,** dar și filtre automate, pentru a crea tablouri care imită rezultatele unor pictori celebri, precum Picasso, etc

Există trei tipuri de tablouri:

Miniaturi, cu lățimea de maxim 10cm

Tablouri, cu lățimea de maxim 57cm, dar nu mai puțin de 19.

Tablouri de perete, cu lățimea de la 1m până la 10m.

Pentru a crea mai ușor imaginile și selecta filtrele potrivite, orice tablou va reține **lățimea** și va conține **rația dintre lățime și înălțime**. Rația va fi întotdeauna între 1 și 3 pentru miniaturi, între 0.618 și 1.618 pentru tablouri, și între 1 și 10 pentru tablourile de perete.

Cerințe:

1. Implementați clase pentru toate tipurile de tablouri, dar și o clasă de bază, care va fi abstractă.
   1. În constructori veți include teste și veți arunca (throw) un întreg/ o eroare dacă nu sunt respectate condițiile.
2. Veți implementa o funcție cantitateUlei() care va calcula de cât ulei este nevoie pentru a crea tabloul, după formula

**cantitateUlei = ariaTabloului \* 0.1mL/cm3**

Metodă se va afla în clasa părinte și va folosi o funcție virtuală utilă, pentru a simplifica calcularea formulei.

Hints:

1. Fiecare tablou are lățimea salvată clar în centimetri, dar e nevoie de rație pentru a calcula lungimea. Puteți folosi întrebările de mai sus ca să vedeți cum implementați

2. Se vor implementa teste (cu try-catch preferabil, dar putem și cu if-uri unde se poate, precum la setarea rației). Cum ele diferă în funcție de tipul de tablou, este întărită utilitatea moștenirii.

## 3. Cum luăm note bune la Colocviu

*Vom folosi un colocviu simplificat, în care am micșorat numărul de clase cerute și păstrat doar cerințe esențiale.*

**Enunţ 2021**

Este 2021 și pentru că 2020 nu se putea termina altfel, calculatorul cuantic al lui Moș Crăciun a suferit câteva pagube. Spiridușii lucrează de zor pentru a îl repara, însă au nevoie de ajutor. Pentru a reuși să salveze Crăciunul, toți spiridușii s-au aliat pentru a pune pe picioare tehnologia care pentru noi încă nu este accesibilă. Doamna Crăciun are grijă de reni și face ordine între jucării, iar bătrânul cu barbă albă nu se poate descurca deloc singur. Știe că studenții de la Facultatea de Matematică și Informatică din București sunt pricepuți și îl pot ajuta cu o soluție temporară astfel încât să gestioneze din timp comportamentul copiilor și dorințele acestora. Cu un produs software bine realizat, Moș Crăciun ar putea să se pună pe treabă încă de pe acum. Bătrânul cunoaște toți copiii, însă trebuie să administreze totul cât mai bine. Trebuie să creați un program care permite evidența tuturor copiilor și a tuturor jucăriilor.

Știm că moșul împarte jucării ce pot fi de mai multe tipuri, iar copiii pot fi copii cuminți, respectiv neastâmpărați.

* Cunoaștem faptul că **toate jucăriile** au o *denumire*, o *dimensiune* (valoare numerică ce reprezintă volumul cutiei în care a fost ambalat), și un *tip* (exemplu: “figurină”, “papușă”, “roboțel”, “telefon”, etc).
* Jucăriile pot fi clasice, educative, electronice sau moderne.
* În cazul celor **clasice** este reținut pe lângă ceea ce este precizat mai sus și materialul și culoarea.
* Pentru jucăriile **educative** se reține abilitatea dezvoltată (exemplu: “gândirea”, “atenția”, “simțul umorului”, “lucru în echipă”, “general”, etc).
* Pentru jucăriile **electronice** se cunoaște numărul de baterii.
* Jucăriile **moderne** pot fi considerate atât educative cât și electronice. În cazul jucăriilor moderne moșul reține în plus brandul și modelul, iar numărul de baterii este 1, respectiv abilitatea dezvoltată este “generală”.
* **Pentru toți copiii,** moșul cunoaște un ID ce este incrementat în mod automat, **numele, prenumele, adresa, vârsta, numărul de fapte bune și jucăriile pe care le primește** (care pot să nu existe sau să fie oricât de multe, în funcție de numărul de fapte bune). Fiecare copil primește jucării în funcție de numărul de fapte bune (dacă numărul de fapte bune este x, copilul primește x jucării).
* Copiii cuminți primesc și un număr de dulciuri.
* Copiii neastâmpărați primesc și un număr de cărbuni.
* Pentru orice copil cuminte, numărul de fapte bune este cel puțin 10, în caz contrar, programul realizat trebuie să afişeze un mesaj corespunzător şi să îi permită lui Moş Crăciun să reintroducă valoarea.

Programul nostru final trebuie să permită rezolvarea problemei descrise mai sus astfel:

1. Să permită memorarea, citirea și afișarea a n copii (indiferent că sunt cuminți sau neastâmpărați).
2. Să permită pentru fiecare copil memorarea, citirea și afișarea a m jucării (indiferent de tipul acestora)
3. Să permită Moșului să găsească un copil după nume (dacă sunt mai mulţi copii cu nume similar, vor fi afişaţi toţi cei care au textul căutat în componenţa numelui lor)
4. Să permită Moșului să ordoneze copiii după numărul de fapte bune

**(Am pus doar 2/5 cerințe dintre opțiunile meniului interactiv)**

**Alte aspecte de implementat:**

* Acţiunile de mai sus vor fi evidenţiate într-un **meniu**.
* Pentru citirea obiectelor este necesar să fie afişate mesaje corespunzătoare informaţiilor citite astfel încât utilizatorul să aibă cunoștințe despre datele pe care le citeşte.
* Pentru afişarea obiectelor este necesar să fie afişate mesaje corespunzătoare informaţiilor afişate.
* Pentru tratarea erorilor/ excepţiilor este necesar să oferiţi mesaje în consolă.

# Pașii de rezolvare

## 

## *// 1. Citim cerința și cream ierarhia de clase. Vom moșteni inclusiv IoBase pentru fiecare clasă de bază*

Pentru a fi ușor de urmărit moștenirile, prima data vom scrie clasele pe care le-am găsit și vom crea inclusiv moștenirile

## *// 2. Adăugăm datele în fiecare dintre clasă*

Reluăm cerința pentru a adăuga toate datele necesare

## *// 3. Adăugăm funcțiile virtuale (2 și 3 pot fi făcute în același timp)*

Este util să reluăm încă o dată ca să adăugăm și metode virtuale unde ar fi utile (e.g. este o funcție care diferă de la clasă moștenitoare la clasă moștenitoare)

## *// 4. Generăm tot ce putem genera -> minim constructori*

AIci este simplu, deoarece avem inclusiv functions.h inclus :), deci modul în care îi generează editorul este suficient.

## *// 5. Facem override la toate funcțiile virtuale NECESARE*

Ctrl+O în fiecare clasă

**Dacă o funcție are implementarea identică cu cea din părinte, nu va fi nevoie :D**

## *// 6. Implementăm citire a n obiecte din cel mai de bază tip*

*POATE sa difere, si vom citi de fapt un obiect mare care compune alte clase.*

E.g. cerinta minimală să fie citirea unei Companie care conține Angajati de diverse tipuri si Job-uri care sunt sau nu sunt ocupate)

## *// 7+ avem cerințe specifice, pe care le vom implementa într-un meniu special. TO BE DETERMINED*

## 