Pràctica 1. Programació orientada a objectes amb C++. Herència i polimorfisme

1. Introducció

Objectius: Familiaritzar-se amb el llenguatge C++ i amb NetBeans 8.2. Entendre els conceptes d'herència i polimorfisme i la seva realització en C++.

Temes de teoria relacionats amb la pràctica: Tema 2. C++ i TADs

Les pràctiques es faran amb l'IDE NetBeans 8.2 i la versió 11 de C++.

2. Enunciat

Aquesta pràctica consta de 7 exercicis i estan ordenats segons la seva dificultat.

Exercici 1.

Crea un programa **main.cpp** que demani el nom a l'usuari, el saludi pel seu nom i li demani una opció de les mostrades a un menú. Aquest menú l'anirem ampliant durant la pràctica, de moment, només tindrà dos opcions; "*Sortir*" o "*Benvinguda*". Si escollim *Sortir* el programa finalitzarà i si escollim *Benvinguda*, el programa ens donarà la benvinguda personalitzada a l'assignatura d'estructura de dades i ens tornarà a mostrar el menú per tornar a escollir una opció.

Per exemple, el programa hauria de mostrar el següent:

```
Hola, com et dius? Anna
Hola Anna, que vols fer?
1. Sortir
2. Benvinguda
2
Benvingut/da a l'assignatura d'estructura de dades Anna
Hola Anna, que vols fer?
1. Sortir
2. Benvinguda
1
Fins a la propera Anna
```

Per fer-ho cal:

a. Al main, definir les variables per guardar el nom, l'opció escollida (integer) i un array de strings per emmagatzemar les diferents opcions del nostre programa. Aquest últim ara serà: string arr_options[] = {"Sortir", "Benvinguda"};
b. Preguntar el nom i agafar-lo fent servir les comandes cout i cin de C++. Caldrà fer l'include de la llibreria *iostream* i el namespace corresponent.
c. Definir una estructura do{...}while(option != 1), que saluda a l'usuari, mostra les opcions del menú amb un bucle for i agafa l'opció l'escollida per l'usuari, comprovant que aquesta sigui una opció vàlida.

Un cop creat aquest programa executa'l. Prova de posar un punt de control i fes el debug. Quins tipus de fitxers tens a la carpeta del teu ordinador del projecte? A què correspon cada tipus?

Exercici 2.

Copia l'anterior main i fem algunes modificacions i ampliacions.

a. Amplia el menú amb l'opció "Redefinir el nom", que modifica el nom de l'usuari entrat a l'inici del programa.
b. El tipus de dades Vector té moltes 'built-in funcions' com ara vector::size() que ens ajuden a tractar-lo. Transformeu l'array de strings arr_options[] a un vector de strings i utilitzeu aquest per mostrar el menú utilitzant la funció vector::size() per saber quan s'ha de parar d'iterar.
c. Un cop funcioni el canvi anterior, agafeu la part de mostrar el menú i la comprovació de que l'opció entrada és correcte i traslladeu tot el codi a un mètode que retorna l'opció escollida.
d. Al main, un cop recollida l'opció escollida per l'usuari, mitjançant una estructura switch, fer una de les tres opcions següents: 1 sortir, 2 Benvinguda o 3 Redefinir el nom.

Digues els dubtes que hagis tingut i explica com els has solucionat.

A continuació teniu un exemple de l'entrada/sortida del codi.

```
Hola, com et dius? Anna
Hola Anna, que vols fer?
1. Sortir
2. Benvinguda
3. Redefinir nom
2
Benvingut/da a l'assignatura d'estructura de dades Anna
Hola Anna, que vols fer?
1. Sortir
2. Benvinguda
3. Redefinir nom
3
Hola, com et dius? Aina
Hola Aina, que vols fer?
1. Sortir
2. Benvinguda
3. Redefinir nom
1
Fins a la propera Aina
```

Exercici 3.

Defineix una classe `Square` amb tres mètodes: `getArea`, `getPerimetre`, i `print` al fitxer **Square.cpp** (i el corresponent **Square.h**). Al programa **main.cpp** les opcions del menú seran: "*Sortir*" i "*Introduir quadrat*".

"*Introduir quadrat*" cridarà des del switch a un mètode que demanarà a l'usuari les dades d'un quadrat (la seva base), crearà l'objecte i farà un print i mostrarà la seva àrea i el seu perímetre per pantalla. El print imprimirà per pantalla que és un `Square(num)`, on num la base del quadrat que s'hagi creat. Un cop hagi acabat tornarà a mostrar el menú.

A més a més, crea un comptador de quadrats al main i passa'l per referència a la funció "*Introduir quadrat*" per que incrementi el valor del punter que rep. Per exemple, el programa hauria de mostrar el següent:

```
Hola, que vols fer?
1. Sortir
2. Introduir quadrat
2
Quadrat 1
Base? 5
Square(5)
L'àrea d'aquest Quadrat és de 5
El perímetre d'aquest Quadrat és de 20

Hola, que vols fer?
1. Sortir
2. Introduir quadrat
2
Quadrat 2
Base? 7
Square(7)
L'àrea d'aquest Quadrat és de 7
El perímetre d'aquest Quadrat és de 28

Hola, que vols fer?
1. Sortir
2. Introduir quadrat
1
Fins a la propera
```

Un cop acabat el codi anterior, pots millorar la classe **Square.cpp** per a què si el valor de la base és 0 o un nombre negatiu enviï una excepció i es mostri per pantalla el text "Atenció: aquest valor no és acceptat". El programa `main.cpp` haurà d'interceptar aquesta excepció i tornar a mostrar el menú.

Al laboratori us explicarem amb més detalls excepcions. A continuació us mostrem un exemple d'una funció per comparar dos nombres. En el vostre cas, l'excepció s'ha de llançar (`throw`) a la classe **circle.cpp**. Noteu que s'ha d'incloure una llibreria: `stdexcept`. Aquesta llibreria inclou moltes excepcions, una d'elles és `invalid_argument`. Vegeu tota la informació a: http://www.cplusplus.com/reference/stdexcept/

Noteu que quan el valor d'a o b és negatiu, es llança l'excepció amb el missatge que es vulgui.

```
#include <stdexcept> // aquest include es necessari al vostre codi
```

```
int compare( int a, int b ) {
    if ( a < 0 || b < 0 ) {  // condició per decidir llançar l'excepció
        throw std::invalid_argument( "received negative value" );// llançar-la
    }
}
```

A continuació teniu el tros del codi main on es crida al mètode `compare` i per recollir l'excepció es fica dins d'una estructura `try … catch`. El `try` conté el tros de codi que crida al mètode que pot llançar l'excepció i el `catch` recull l'excepció si es produeix. És molt important que veieu que SEMPRE es recull l'excepció per referència (per això té un & a l'objecte e).

```
try {
    compare( -1, 3 ); // Aqui vindrà el vostre codi del main
}
catch( const std::invalid_argument& e ) {
    // codi de gestió de l'excepció, en aquest exercici: missatge per consola
}
```

Quan implementeu el control de nombres negatius, per exemple, el programa hauria de mostrar el següent:

```
Hola, que vols fer?
1. Sortir
2. Introduir quadrat
2
Quadrat 1
Base? -1
S'ha produït una excepció

Hola, que vols fer?
1. Sortir
2. Introduir quadrat
2
Quadrat 1
Base? 0
S'ha produït una excepció

Hola, que vols fer?
1. Sortir
2. Introduir quadrat
1
Fins a la propera
```

Exercici 4.

Defineix la classe `Rectangle` amb tres mètodes: `getArea`, `getPerimetre`, i `print` al fitxer **Rectangle.cpp i Rectangle.h**. La classe Rectangle guardarà la base i l'alçada (ambdós de tipus float). Modifica la funció de "*Introduir quadrat*" de l'exercici anterior per la funció "*Afegir figura*", què afegirà un quadrat o un rectangle segons com sigui l'entrada de l'usuari i farà un print, mostrarà la seva àrea i el seu perímetre.

Per crear un quadrat l'usuari haurà d'introduir una S, on S indica que la figura és un quadrat. Per crear un rectangle l'usuari haurà d'entrar una R, on R indica que la figura és un rectangle. Recorda tractar les excepcions per valors d'entrada negatius o cero per cada figura i portar un comptador per ambdues figures.

Crea també una altre opció en el menú ("*Glossari de figures*") per veure quants quadrats i rectangles s'han creat des de l'inici del programa. Per exemple, el programa hauria de mostrar el següent:

```
Hola, que vols fer?
1. Sortir
2. Afegir figura
3. Glossari de figures
2
(S)quare/(R)ectangle? S
Quadrat 1
Base? 3
Square(3)
L'àrea d'aquest Quadrat és de 9
El perímetre d'aquest Quadrat és de 12

Hola, que vols fer?
1. Sortir
2. Afegir figura
3. Glossari de figures
2
(S)quare/(R)ectangle? R
Rectangle 1
Base? 4
Altura? 3
Rectangle(4, 3)
L'àrea d'aquest Rectangle és de 12
El perímetre d'aquest Rectangle és de 14

Hola, que vols fer?
1. Sortir
2. Afegir figura
3. Glossari de figures
3
Tens 1 quadrats i 1 rectangles.

Hola, que vols fer?
1. Sortir
2. Afegir figura
3. Glossari de figures
1
Fins a la propera
```

Exercici 5.

Fes una opció més en el menú de l'exercici anterior per tal de fer que llegeixi les dades d'un fitxer i les vagi processant fins a arribar al final. Recorda seguir tractant les excepcions i portar el comptador de les figures per mostrar-ho si es demana l'opció de l'exercici anterior "*Glossari de figures*". Per exemple, en el fitxer hi haurà la següent informació:

```
R 2 3
S 1
S 2
R 1 1
R 3 2
```

On S fa referència a un Square i R a un Rectangle. Tenint en compte el fitxer anterior, el programa en execució generarà:

```
Hola, que vols fer?
1. Sortir
2. Afegir figura
3. Llegir fitxer
4. Glossari de figures
3
Ruta al fitxer:
Figures.txt
Reading file Figures.txt...
Rectangle(2, 3)
L'area d'aquest Rectangle és de 6
El perímetre d'aquest Rectangle és de 10
Square(1)
L'area d'aquest Quadrat és de 1
El perímetre d'aquest Quadrat és de 4
Square(2)
L'area d'aquest Quadrat és de 4
El perímetre d'aquest Quadrat és de 8
Rectangle(1, 1)
L'area d'aquest Rectangle és de 1
El perímetre d'aquest Rectangle és de 4
Rectangle(3, 2)
L'area d'aquest Rectangle és de 6
El perímetre d'aquest Rectangle és de 10

Hola, que vols fer?
1. Sortir
2. Afegir figura
3. Llegir fitxer
4. Glossari de figures
4
Tens 2 quadrats i 3 rectangles.
```

```
Hola, que vols fer?
1. Sortir
2. Afegir figura
3. Llegir fitxer
4. Glossari de figures
2
(S)quare/(R)ectangle? R
Rectangle 4
Base? 5
Altura? 6
Rectangle(5, 6)
L'area d'aquest Rectangle és de 30
El perímetre d'aquest Rectangle és de 22

Hola, que vols fer?
1. Sortir
2. Afegir figura
3. Llegir fitxer
4. Glossari de figures
4
Tens 2 quadrats i 4 rectangles.

Hola, que vols fer?
1. Sortir
2. Afegir figura
3. Llegir fitxer
4. Glossari de figures
1


Fins a la propera
```

Exercici 6.

Crea un projecte amb diverses classes relacionades amb geometria:

a. El quadrat i el rectangle són casos particulars de quadrilàter. Reutilitza el que puguis del projecte anterior, conservant les quatre opcions de menú que ja tenim, però fes que la classe `Square` i `Rectangle` heretin de la classe `Quadrilateral`, per exemple class Square: public Quadrilateral. Fes que el constructor de cada classe al cridar-se imprimeixi per pantalla "Constructor de X", on X és el nom de la classe (Quadrilateral, Square o Rectangle) i pensa que quan estiguis creant un Square cridaràs també al constructor de Quadrilateral que també rep paràmetres. Assegura't que quan facis un Square o Rectangle s'executen els dos constructors.
b. Comprova que totes les funcions del menú funcionen correctament inclosa la de llegir les figures, fixa't que per cada una es mostrin per pantalla els corresponents constructors.
c. Fes que el mètode `getArea()`, `getPerimeter()` i `print()` de Quadrilateral sigui virtual.

Exercici 7.

Fes una còpia del projecte anterior i agrega-hi la classe **QuadrilateralContainer**, que serà un vector (`std::vector<Quadrilateral*> v`) de com a màxim 10 figures que poden ser tan Square com Rectangle, amb els següents mètodes:

a. `void addQuadrilateral (Quadrilateral *)` que afegirà al vector la figura que se li passi per paràmetre, recorda que d'aquesta manera també podràs afegir Square i Rectangle.
b. `float getAreas()` que retornarà la suma de les àrees dels quadrats i rectangles continguts en el vector. Per recórrer el vector usa un iterador.
c. En el **main.cpp** crea un objecte `QuadrilateralContainer` i afegeix cada figura que estiguis creant tant directament com des del fitxer, quan escollis l'opció corresponent en el menú.
d. Fes una opció més al menú per calcular la suma de totes les àrees i mostrar-la.
e. Comprova que es criden tots els constructors i destructors de les classes. Per comprovar-ho, posa un missatge al destructor de cada classe, per saber quins destructors es criden en tot moment a l'acabar el programa.

A partir del què has observat respon:

a. Què ens permet fer l'herència que no podríem fer altrament?
b. Que passa si `getArea()` de la classe Quadrilateral no és virtual? Perquè?
c. Perquè els constructors i destructors els hem de cridar a les classes derivades i no a la classe base?
d. Es podria fer que `getArea()` i `getPerimetre()` fos només un mètode de la classe "Quadrilateral"?
e. Anomena els membres de dades definits en els teus programes i digues a quina classe pertanyen. Explica les decisions de visibilitat de cadascun
f. L'iterador que recorre les figures, quant s'incrementa cada cop? Perquè?

3. Lliurament

A partir de la descripció del problema, es demana:

- Implementar els exercicis en C++ i usar el **NetBeans 8.2 amb C++ versió 11**. Lliurar el codi C++ corresponent als vostres exercicis en una carpeta anomenada *codi*, amb una subcarpeta per a cada exercici. Els comentaris de cada exercici (observacions, decisions preses i resposta a les preguntes plantejades, si n'hi ha) els fareu com a comentari al fitxer `main.cpp` de cada exercici.

Com a màxim el dia del lliurament es penjarà en el campus virtual un fitxer comprimit **en format ZIP** amb el nom del grup (A, B, C, D, E o F), el nom i cognoms de la persona i el número de la pràctica com a nom de fitxer. Per exemple, **GrupA_BartSimpson_P1.zip**, on P1 indica que és la "pràctica 1". El fitxer ZIP inclourà: la carpeta amb tot el codi.

Els criteris per acceptar la pràctica són:

- La pràctica ha de funcionar en la seva totalitat.
- La pràctica ha de ser orientada a objectes.

IMPORTANT: La còpia de la implementació de la pràctica implica un zero a la nota de pràctiques de l'assignatura per les persones implicades (tant la que ha copiat com la que ha deixat copiar).

4. Planificació

Per fer aquesta pràctica disposeu de dos setmanes.

Els professors us proposem la següent planificació:

- **Setmana 1 (del 15 de febrer al 19 de febrer)** → laboratori (Lab1)
  - Implementació de l'exercici 1 a l'exercici 5
  - No fareu lliurament aquesta setmana, només la última setmana en aquesta pràctica
- **Setmana 2 (del 22 de febrer al 26 de febrer)** → laboratori (Lab2)
  - Implementació de l'exercici 6 a l'exercici 7
  - Cal presentar el codi de tots els exercicis

El lliurament final de la pràctica serà el **dia 7 de Març de 2020** al campus virtual.