Programación 2009

Tarea 13

Las preguntas precedidas por un asterísco son para los alumnos de maestría. En licenciatura, dan puntos extras.

Ejercicio 1 *Ilustrar a través de un ejemplo de su elección que lo que dijimos en la clase de los métodos de una clase heredando de otra vale también para* ***métodos estáticos*** *: (1) los métodos redefinidos en la clase hija tienen* ***preeminencia sobre los de la clase madre*** *y (2) si hay sobrecargos de este método en la clase madre, las versiones sobrecargadas son* ***escondidas****.*

Ejercicio 2 *Para cada de las clases Cpriv, Cprot, Cpubl, decir que datos de B pueden manipular y como los datos de B aparecen al exterior (¿son accesibles o no?)*

```
class B                        {
        private:
                int x;
        protected:
                int y;
        public:
                int z;
};
class Cpriv : private   B {
...
};
class Cprot : protected B {
...
};
class Cpubl : public    B {
...
};
```

Ejercicio 3 *Definamos cuatro clases C0, C1 heredando de C0, C2 heredando de C1, C3 heredando de C1. Definamos 4 objetos o0, o1, o2 y o3 de cada uno de estas clases. Decir si en cada caso, la instrucción es válida y por qué :*

```
1. C3 &aptr = &o1;
2. C0 &aptr = o3;
3. C3 *aptr = &o2;
4. C1 *aptr = &o2; C0 **dptr= &aptr;
5. C1 &aref = o2;  C2 &adref = aref;
5. C1 &aref = o2;  C0 &adref = aref;
```

Ejercicio 4 *Escribir un programa que ilustra que al llamar una función virtual dentro de un constructor se llama necesariamente la versión local a la clase.*

***Ejercicio 5** *Verificar en el código siguiente que el compilador no acepta una conversión directa (en la cuarta linea del main) entre dobles apuntadores, y que por eso se utilizo un reinterpret_cast. Ahora, imprimir un mensaje específicos a ambos métodos declarados, y deducir del comportamiento del programa en que puede ser muy peligroso hacer este cast (de la cuarta linea del main) y que por eso el compilador no lo autoriza de manera directa.*

```
class Caja {
};
class CajaRegalo : public Caja {
public:
  virtual void abrirRegalo();
};
class CajaBomba : public Caja {
public:
  virtual void prenderBomba();
};
int main() {
  CajaRegalo c;
  CajaRegalo *cptr = &c;
  CajaRegalo **cptrptr = &cptr;
  Caja **jptrptr = cptrptr;
  CajaBomba  b;
  CajaBomba *bPtr = &b;
  *jptrptr = bPtr;
  cptr-> abrirRegalo();
}
```