Departament de Disseny i Programació de Sistemes Electrònics
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Prova final de TECPRO

15 de juny de 2012 — Enginyeria de Sistemes TIC — 150 MIN

COGNOMS:

NOM: GRUP de LAB:

Problemes curts

Exercici 1 [1 punt]. Si x i y són dues instàncies de la classe C, quina de les següents afirmacions és certa?

- ◯ x i y sempre tenen els mateixos atributs.
- ◯ Per comparar x i y amb una expressió com x==y cal que la classe C tingui definit el mètode __eq__.
- ◯ x i y sempre tenen els mateixos mètodes.
- ◯ En ser una instància, x no es pot escriure usant **print**.

Exercici 2 [1 punt]. El mètode __init__ d'una classe inicialitza les instàncies d'aquesta classe. Quina de les següents afirmacions és certa?

- ◯ Si a una classe li manca el mètode __init__ és que les instàncies de la classe no tenen atributs.
- ◯ Cal que el mètode __init__ tingui un paràmetre per a cada atribut de l'objecte.
- ◯ Si una classe A és subclasse de B, A no pot tenir mètode __init__.
- ◯ El mètode __init__, com a molt, s'aplica una sola vegada a cada instància.

Exercici 3 [1 punt]. Quina de les següents afirmacions és certa?

- ◯ Dues classes diferents no poden tenir la mateixa superclasse.
- ◯ Si a és una instància de la classe A i b una instància de la classe B, des d'un mètode d'a no es pot invocar un mètode de b.
- ◯ En un programa `Python` podríem trobar expressions com aquesta: x==a.b().c.d().
- ◯ Quan una funció rep com argument una instància a d'una classe A, des del cos de la funció no es pot modificar el valor d'a.

Exercici 4 [1 punt]. Si a i b són variables que referencien instàncies de la mateixa classe Exemple, quina de les afirmacions següents és certa?

- ◯ a==b valdrà **True** només quan les instàncies referenciades per a i b siguin la mateixa instància.
- ◯ a**is**b valdrà **False** si i només si les instàncies referenciades per a i b són la mateixa instància.
- ◯ a**is**b valdrà **True** o **False** segons com s'hagi implementat la classe Exemple.
- ◯ Cap de les anteriors afirmacions és certa.

Exercici 5 [1 punt]. Considera el següent programa, que defineix algunes classes i, finalment escriu el valor d'una expressió:

```
class S(object):
    def __str__(self):
        return str(self.x())

class A(S):
    def x(self):
        return str(B())

class B(S):
    def __init__(self,d=4):
        self.d = d

    def x(self):
        return self.d

print A()
```

Quina de les següents afirmacions sobre l'exemple anterior és cert?

- ◯ L'exemple acaba escrivint 4.
- ◯ L'exemple és erroni ja que nou es pot usar una classe com a argument.
- ◯ L'exemple és erroni atès que la classe A no té __init__.
- ◯ L'exemple és erroni ja que es crida A() sense passar cap argument.
- ◯ Cap de les respostes anteriors és correcta.

Exercici 6 [1 punt]. Considera el següent fragment de programa, que defineix algunes classes i, finalment executa una sèrie de càlculs:

```
class Crash(Exception):
    pass

class X(object):
    def m1(self, p):
        try:
            if p > 12:
                raise Crash('Granproblema')
            else:
                return p * 3
        except OutOfMem:
            return p + 2

o = X()
try:
    x = o.m1(5)
    x += o.m1(x)
except OutOfMem:
    print "Whatthehell"
print x
```

Quina de les següents afirmacions sobre aquest petit programa és certa?

- ◯ El programa avortarà l'execució i escriurà `'Granproblema'`.
- ◯ El programa acabarà l'execució correctament i escriurà `'Whatthehell'`.
- ◯ El programa té errors sintàctics i no es podrà executar.
- ◯ El programa acabarà i escriurà 32.

Exercici 7 [1 punt]. Considereu un edifici de pisos i el seu model UML. La millor forma de representar la relació entre un pis i una planta en el model UML seria:

- ◯ Una composició.
- ◯ Una agregació.
- ◯ Una associació unidireccional.
- ◯ Una associació bidireccional.

Exercici 8 [1 punt]. Dissenyeu una funció recursiva `p(x)`, on `x` és un natural, que torni la suma dels dígits de `x`. Per exemple, si `x` és 371, la funció hauria de tornar 11. Només podeu usar càlcul sobre enters.

Exercici 9 [1 punt]. En un cert computador el programa A obeeix a la funció de cost en cas pitjor $T_A(n) = 2n^2 + 21$ i el programa B es caracteritza per tenir una funció de cost en cas pitjor $T_B(n) = 31n + 4$. Aleshores, suposant sempre que les dades són les pitjors possibles, a partir de quin valor d'n el programa B és més ràpid?

- ◯ B sempre és el més ràpid per que és $O(n)$ mentre que A és $O(n^2)$.
- ◯ B és el més ràpid per $n \geq 15$.
- ◯ B és el més ràpid per $n \geq 35$.
- ◯ Cap de les anteriors és certa.

Exercici 10 [1 punt]. Suposeu que sobre una pila fem push dels enters entre 0 i 9 de forma ordenada. Entre mig de les operacions de push intercalem arbitràriament operacions de pop que escriuen el valor que treuen de la pila. Quina de les següents seqüencies escrites pels pop és impossible?

◯ 4 3 2 1 0 9 8 7 6 5

◯ 4 6 8 7 5 3 2 9 0 1

◯ 2 5 6 7 4 8 9 3 1 0

◯ 4 3 2 1 0 5 6 7 8 9

Exercici 11 [1 punt]. Escriviu una funció tal que, donats dos arbres binaris, determini si són iguals o no.

Exercici 12 [1 punt]. Recordeu que un enter `Python` habitualment està format per quatre bytes: $b_3b_2b_1b_0$. Escriviu una expressió `Python` que, donat un enter `i`, obtingui el valor del segon byte b_1.

Problema llarg

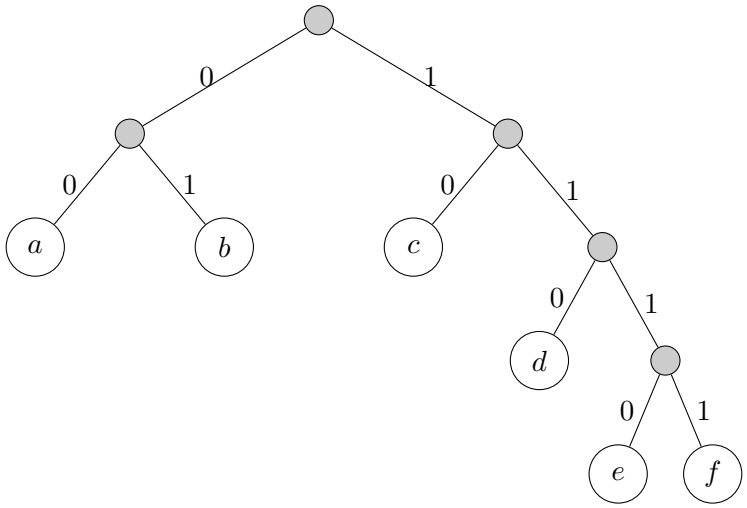
Exercici 13 [8 punts]. Considereu que un alfabet A és un conjunt de lletres. Per exemple $A = \{a, c, e\}$ és un alfabet. Una paraula de l'alfabet A és una seqüència d'elements d'A, *aaacee*, per exemple, és una paraula de l'alfabet A.

La codificació de Huffman representa cada lletra d'un alfabet amb una seqüència de bits. Per exemple, una codificació Huffman de A podria ser aquesta: $a = 0$, $c = 10$, $e = 11$. Aquesta codificació té diverses propietats:

1. Les lletres menys freqüents (que surten menys) tenen codis més curts.
2. El codi té la propietat del *prefix*: el codi d'una lletra no és mai el prefix del codi d'una altra lletra.

A causa de la propietat del prefix, podem representar paraules d'un alfabet simplement concatenant els seus codis. Per exemple, la paraula *aaacee* es representaria aplicant el codi anterior com 0001011. Fixeu-vos que, per la mateixa propietat, és molt fàcil descodificar-la i obtenir la paraula inicial.

Els codis Huffman d'un alfabet concret es representen habitualment usant un arbre binari. En aquest arbre, les fulles representen les lletres de l'alfabet i les arestes descendents d'un node s'etiqueten amb 0 (fill esquerra) i 1 (fill dret). La següent figura representa un arbre de Huffman per a l'alfabet $B = \{a, b, c, d, e, f\}$:



Es demana que dissenyeu una funció tal que, donat un arbre de codificació Huffman i una llista d'enters (0 o 1), retorni la cadena de caràcters que s'obté de descodificar la llista segons el codi de l'arbre. A tal efecte podeu assumir que l'arbre binari és un objecte de classe ABin i que disposa dels mètodes es_fulla(), valor(), fill_d() i fill_e() amb el seu significat habitual. En particular, quan un node e és fulla, llavors tant e.fill_d() com e.fill_e() retornen **None**.