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

Prova final de TECPRO

21 de juny de 2011 — Enginyeria de Sistemes TIC — 150 MINUTS

COGNOMS:

NOM: GRUP de LAB:

PART FIXA

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

- ◯ Un mòdul pot contenir diverses classes.
- ◯ Un mòdul és el mateix que un fitxer.
- ◯ La definició d'una classe pot abastar diversos mòduls.
- ◯ Cap de les anteriors afirmacions és certa.

Exercici 2 [1 punt]. En el cas de les instàncies d'una classe Python, quina de les afirmacions següents és certa?

- ◯ Poden tenir atributs públics i privats.
- ◯ Tots els atributs són públics.
- ◯ Tots els atributs són privats.
- ◯ Cap de les anteriors afirmacions és certa.

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

- ◯ Si a la classe A li manca el mètode __init__, llavors no poden existir instàncies de la classe A.
- ◯ Si la classe A és una subclasse de B i B és una subclasse de C aleshores tots els mètodes d'una instància de la classe C també els té una instància de la classe A.
- ◯ Si C és la super-classe d'A i B, llavors les instàncies d'A i de B tenen els mateixos mètodes.
- ◯ Si dues instàncies a i b tenen els mateixos mètodes, llavors és que són de la mateixa classe.

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 True 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 m1(self,p):
        return self.x('a') + p

class A(S):
    def x(self, p):
        return 2*p

class B(S):
    def x(self, p):
        return 3*p

p = A(S)
print p.m1('b')
```

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

- ◯ L'exemple és erroni ja que dues classes no poden tenir un mètode amb el mateix identificador.
- ◯ L'exemple és erroni ja que el mètode m1 se la classe S crida al mètode x i la classe S no té cap mètode anomenat x.
- ◯ L'exemple acaba escrivint aab.
- ◯ L'exemple acaba escrivint aaab.

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

```
class Problema(Exception):
    pass

class A(object):
    def m1(self, p):
        if p > 12:
            raise Problema('Gran problema')
        else:
            return p * 3

o = A()
try:
    x = o.m1(5)
    x += o.m1(x)
except Exception:
    print 'Too much for me'
```

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

- ◯ El programa avortarà l'execució i escriurà `'Gran problema'`.
- ◯ El programa acabarà l'execució correctament i escriurà `'Too much for me'`.
- ◯ El programa té errors sintàctics i no es podrà executar.
- ◯ Cap de les afirmacions anteriors és certa.

Exercici 7 [1 punt]. Si en el context d'un model UML, la classe A té una associació unidireccional amb la classe B, és a dir $A \to B$, quina de les afirmacions següents és certa?

- ◯ Una instància de la classe B no està mai relacionada amb una instància de la classe A.
- ◯ Una instància de la classe A sempre està relacionada amb una instància de la classe B.
- ◯ Donada una instància de la classe A podem accedir directament a les instàncies de la classe B amb que es relaciona.
- ◯ Cap de les afirmacions anteriors és certa.

Exercici 8 [1 punt]. Dissenyeu una funció recursiva p(x), o x és un natural, que torni True ssi x és un parell. Considereu x==0 i x==1 com els casos base.

Exercici 9 [1 punt]. En un cert computador el programa A obeeix a la funció de cost en cas pitjor $T_A(n) = 2n^3 - n + 21$ i el programa B es caracteritza per tenir una funció de cost en cas pitjor $T_B(n) = 2n^3 + n - 2000$. Aleshores, quina de les següents afirmacions és certa?

- ◯ Per a qualsevol joc de dades C que conté n elements, el cost d'aplicar l'algoritme B és de $2n^3 + n - 2000$.
- ◯ Com des d'un punt de vista asimptòtic A i B tenen cost $O(n^3)$ en cas pitjor, podem assegurar que sigui quina sigui la mida de les dades a tractar, A i B trigaran el mateix temps.
- ◯ Per a dades de mida petita, B trigarà menys temps que A.
- ◯ Cap de les afirmacions anteriors és certa.

Exercici 10 [1 punt]. Què calcula la següent funció, que usa una pila, sabent que el paràmetre n és un natural diferent de zero?

```
def misteri(n):
    s = Stack()
    for i in range(1,n+1):
        s.push(i)
    while len(s) > 1:
        a = s.top(); s.pop()
        b = s.top(); s.pop()
        s.push(a*b)
    return s.top()
```

Exercici 11 [1 punt]. Considereu la següent seqüència d'enters:

$$53, 2, 87, 34, 32, 15, 60, 61, 52$$

Dibuixeu l'arbre binari de cerca que s'obtindria si inseríssiu aquests elements en un arbre buit i en l'ordre en que es troben escrits.

Exercici 12 [1 punt]. Considereu l'enter a l'expressió binària del qual és $a = a_{31}a_{30}\cdots a_2a_1a_0$. Dissenyeu una funció tal que, donat l'enter a i treballant únicament amb enters, retorni el nombre de 1's que té la seva expressió binària.

PART OPTATIVA: Trieu UN dels dos problemes que segueixen

Exercici 13 [8 punts]. Segur que coneixeu el joc del "tres en ratlla". En aquest exercici es demana que dissenyeu i implementeu una classe per representar el tauler de joc del "tres en ratlla" que anomenareu TresER. Aquesta classe assumirà que:

- El tauler de joc té forma matricial amb 3 files i 3 columnes numerades de 0 a 2.
- Té les caselles identificades per les seves coordenades.
- Els dos jugadors s'identifiquen amb els números 0 i 1 respectivament.
- La classe també controla les fitxes que els jugadors no tenen en el taulell.

La classe ha de tenir els següents mètodes:

- __init__(self)

  Crea un tauler inicialitzat de forma que cada jugador té tres fitxes disponibles i cap d'elles és al tauler.

- posa_fitxa(self, jugador, casella)

  Posa la fitxa del jugador jugador en la casella casella. casella és un tuple de coordenades. Si el jugador no té fitxes disponibles o la casella ja estava ocupada, aixeca una excepció.

- treu_fitxa(self, casella)

  Treu la fitxa de la casella casella i la torna al jugador corresponent. casella és un tuple de coordenades. Si la casella no contenia cap fitxa, aixeca una excepció.

- tres_en_ratlla(self, jugador)

  Retorna True si el jugador jugador té tres fitxes en ratlla en el taulell.

- disponibles(self, jugador)

  Retorna el nombre de fitxes de que un jugador disposa per posar al taulell.

La classe hauria de superar un doctest com el següent:

```
>>> t = TresER()
>>> t.disponibles(0)
3
>>> t.posa_fitxa(0, (1,1))
>>> t.posa_fitxa(0, (2,2))
>>> t.disponibles(0)
1
>>> t.posa_fitxa(0, (1,2))
>>> t.tres_en_ratlla(0)
False
>>> t.treu_fitxa( (1,2) )
>>> t.posa_fitxa(0, (0,0))
>>> t.tres_en_ratlla(0)
True
```

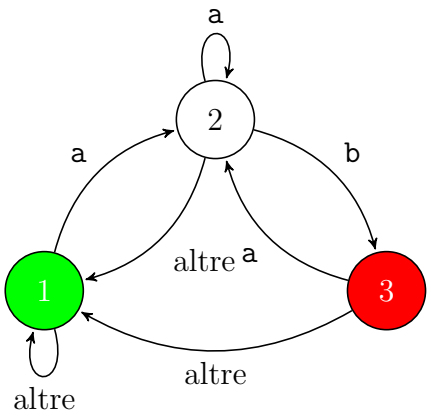
Exercici 14 [8 punts]. Un autòmat finit és un giny que pot usar-se per reconèixer cadenes de caràcters i que està definit per:

1. Un conjunt d'estats $E = \{e_1, \ldots, e_k\}$.
2. Dos estats especials: l'estat inicial $s \in E$ i l'estat final $f \in E$.
3. Una funció de transició δ que indica, donat un estat i un caràcter, quin és el següent estat:

$$\begin{array}{rcl} \delta : E \times C & \longrightarrow & E \\ (e, c) & \mapsto & \delta(e_i, c) = e_j \end{array}$$

Observeu el seu funcionament a través del següent exemple. Imagineu-vos l'autòmat definit pel conjunt d'estats $E = \{1, 2, 3\}$ on l'estat inicial és $s = 1$, l'estat final és $f = 3$ i la funció δ està definida per la següent taula, que també dibuixem en forma de diagrama:

	Entrada	
Estat	Caràcter	Sortida
1	`a`	2
	qualsevol altre	1
2	`a`	2
	`b`	3
	qualsevol altre	1
3	`a`	2
	qualsevol altre	1



Aquest autòmata permet distingir certes cadenes de caràcters. Considerem la cadena `'abcd'`. Per saber si el nostre autòmat la reconeix, simplement comencem en l'estat inicial, 1, i prenem el primer caràcter de la cadena, la `'a'`. Aplicant la funció de transició sabem que el següent estat serà el 2. Considerem el següent caràcter, la `'b'` i apliquem de nou la funció de transició. Ara ens portarà a l'estat 3. Repetiu de nou el procés amb les lletres `'c'` i `'d'`. L'autòmat reconeix la cadena `'abcd'` si després d'aquest procés acaba en l'estat final. Com veieu, la cadena `'abcd'` no és reconeix. En canvi, la cadena `'cdaab'` sí que es reconeix.

En aquest exercici *es demana* que dissenyeu i implementeu la classe Automat. Aquesta classe només té dos mètodes:

- __init__(self, n, t)

  n és un enter que correspon al número d'estats de l'autòmat. Sempre assumim que els estats van de $1, \ldots, n$, i que 1 és l'estat inicial i n l'estat final.

  t és una llista que representa la funció de transició. Aquesta llista està formada per tuples (e1, c, e2) on e1 és un estat, c un caràcter i e2 un altre estat. Un tuple (e1,c,e2) indica que, si estem en l'estat e1 i arriba el caràcter c, llavors hem d'anar a parar a l'estat e2.

  Al caràcter `'.'` li donarem el significat de "qualsevol altre".

- reconeix(self, s)

  Retorna True ssi l'autòmat reconeix s.

Seguint amb l'exemple anterior, aquesta classe hauria de passar el següent doctest:

```
>>> a = Automat( 3, [(1, 'a', 2), (1, '.', 1),
                     (2, 'a', 2), (2, 'b', 3), (2, '.', 1),
                     (3, '.', 1), (3, 'a', 2) ])
>>> a.reconeix('abcd')
False
>>> a.reconeix('cdaab')
True
```