IIC 2143 Ingeniería de Software
Interrogación 2 - Semestre 2/2018

Responda cada pregunta en hoja separada
Entregue una hoja con su nombre para cada pregunta, aunque sea en blanco
Tiempo: 2 horas
Recuerden que se encuentran bajo el código de honor

1. (1.5 pts) Un multicine tiene varias salas de cine; en cada sala se exhibe una película en varias funciones a lo largo del día. Una persona que quiere comprar una entrada para una película determinada en una función determinada va a la boletería del multicine; allí, le venden la entrada sólo si la sala correspondiente no está llena para esa función y la función aún no empieza.
Cuando una persona, con una entrada, llega a la portería del multicine, se le permite entrar sólo si la entrada es para una función que aún no empieza, y si tiene la edad suficiente según la calificación de la película, que puede ser "todo espectador", "mayores de 14", o "mayores de 18".
Cuando una persona, con una entrada, llega a la puerta de una sala del multicine, se le permite entrar sólo si la entrada corresponde a la película que se exhibe en esa sala y a la función que va a empezar, y si la función anterior ya terminó.
Dibuja un modelo de dominio (usando la notación del diagrama de clases de uml) que ilustre las principales clases y relaciones (asociaciones, generalizaciones) entre clases que se desprenden del enunciado. En las clases, incluye atributos; y en las asociaciones, nombres y multiplicidades.

2. (1.5 pts) Una cuenta corriente después de ser creada puede estar en proceso de apertura (hasta que se completen los requisitos) para finalmente quedar abierta y lista para ser usada. Igualmente, la cuenta antes de ser cerrada pasa por un proceso en que por ejemplo se espera que lleguen todos los movimientos pendientes (por ejemplo 1 mes).
Una cuenta abierta permite hace depósitos y giros. Generalmente se permite que la cuenta se "sobregire" (quede negativa) hasta un cierto máximo pactado. Al sobrepasar ese valor, la cuenta se bloquea y ya no es posible realizar mas giros. Una cuenta bloqueada solo puede ser liberada por el ejecutivo de cuentas después que el titular de la cuenta haya depositado de modo de que no quede sobrepasado el monto de sobregiro pactado.
Una cuenta puede ser bloqueada por el ejecutivo en cualquier momento si se detecta actividad "sospechosa"
Modela el comportamiento de un objeto cuenta corriente utilizando un diagrama de estados UML.

3. (1.5 pts) Tu equipo de desarrollo ha realizado un levantamiento de requerimientos funcionales para un software de control de activo fijo de un cliente. Los relatos de usuario se han priorizado y catalogado en 3 niveles de complejidad (se usará una escala exponencial de story points comenzando con 1 punto para los relatos de complejidad baja). La tabla muestra para cada relato, su nivel de complejidad y la prioridad que le ha asignado el *product owner*

Se ha comprometido la entrega en 8 semanas, a partir de la primera semana de Octubre. Se acaba de completar el primer sprint (de 2 semanas) en que el equipo, formado por 5 desarrolladores, logró implementar los relatos US01, US03, US05 y US10.

a) (1 ptos) Haz una planificación para las iteraciones que restan antes del release. En caso que sea necesario hacer ajustes, diseñe una estrategia de negociación basada en ajuste de calendario y otra basada en ajuste por alcance. ¿Cual estrategia es mejor? Justifique su respuesta.

b) (0.5 ptos) Cuantifica tamaño, esfuerzo y duración del proyecto en el supuesto de que se acepte un ajuste por calendario. ¿De acuerdo a la planificación, cual sería el modelo que relaciona esas 3 variables?

Relato	Complejidad	Prioridad
US01	mediana	2
US02	mediana	8
US03	baja	1
US04	mediana	5
US05	alta	4
US06	baja	6
US07	alta	10
US08	mediana	9
US09	alta	7
US10	baja	3
US11	mediana	12
US12	mediana	11
US13	alta	16
US14	mediana	14
US15	mediana	15
US16	baja	13
US17	mediana	17
US18	mediana	18
US19	alta	20
US20	alta	19

4. (1.5 ptos)A continuación código que sirve para implementar un sistema de facturación. El objeto de tipo PaymentSystem mantiene una lista de las ordenes y de los productos que se manejan. La orden incluye una serie de items (uno por cada producto comprado) que deben ser impreso uno por línea.

```
class PaymentSystem
  @@ orders
  @@ products
  def self.process
    orders.each do |order|
     order.printOrder
    end
  end
  def self.getProduct (code)
    products.each |prod|
        if code == prod.code
         return prod
     end
  end
  def self.addProduct(product)
    products << product
  end
  def self.addOrder(order)
    orders << order
  end
end

class Product
  attr_reader :code, :name, :price
  def initialize (code, name, price)
    @code = code
    @name = name
    @price = price
  end
end

class Item
 @productCode
 @quantity
  def printLine
    product = PaymentSystem.getProduct(@productCode)
    amount = product.price * quantity
    puts "#{quantity} #{product.code} #{product.name} #{product.price} #{amount}"
  end
end

class Order
  @items
end

class RetailOrder < Order
  def addItem (code, quantity)
    theItem = Item.new (productCode, quantity)
    @items << item
   end
   def printOrder
     items.each do |item|
       item.printLine
     end
    end
end
```

2	1234	apple	1.20	2.40
1	2134	banana	1.50	1.50
3	6565	melon	2.00	6.00
2	5545	pear	1.50	3.00

a) (0.5 pts)Dibuje el diagrama de clases UML asociado a este código
b) (1 ptos) Dibuje un diagrama de secuencia que muestre como se lleva cabo la impresión de una factura cuando el objeto de tipo RetailOrder recibe un mensaje printOrder