PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

IIC2233 - Programación Avanzada (I/2017)
Tarea 4

1. Objetivos

- Aplicar conocimientos de simulación por eventos discretos.

2. Introducción

Han pasado días desde que terminó el gran RQL pero todos los cálculos llegaban a la misma conclusión: la probabilidad de supervivencia humana es del 0 %. Sin ganas de seguir, descubres una dimensión paralela controlada por el Dr. Mavrakis y te inscribes a un curso llamado Avanzación Programada. Lo único que sabes es que al Dr. Mavrakis no le gustan las personas de otra dimensión, por lo que hará el curso difícil. Intentará hacerte reprobar y que vuelvas a tu dimensión. Para saber si lograrás pasar este nuevo curso, decides hacer una simulación con tus conocimientos de programación.

3. Problema

Para esta tarea, deberás implementar una simulación por eventos discretos (DES) de un semestre en la dimensión del Dr. Mavrakis según cómo era programación avanzada (el curso de tu dimensión). Para lograr este objetivo, debes tomar en cuenta los eventos que podrían ocurrir en el curso, como las tareas, actividades, publicación de notas, ver partidos de fútbol, etc. En particular, interesa observar cómo es la vida de un alumno del curso y sacar algunos datos como porcentaje de aprobación, mejor tarea y muchos más.

4. Integrantes del curso (24 %)

4.1. Alumnos

Todos los alumnos que tomaron Avanzación Programada le dedican una cierta cantidad de horas por semana al curso. Esas horas se gastan en realizar la tarea que tengan y/o estudiar la materia que corresponde a la semana. Además, cuentan con ciertas características únicas que afectan su rendimiento en el curso, es decir, influyen en el *progreso* que el alumno tenga en sus evaluaciones y por lo tanto en la nota que obtienen. Estas características son: *confianza*, *manejo de contenidos*, *cantidad de créditos tomados* y *nivel de programación*.

Por último, los alumnos tienen distintas personalidades que influyen en el *manejo de contenidos*, *nivel de programación* o el *progreso* que logra en una evaluación.

4.1.1. Cantidad horas de dedicadas al curso H

Los alumnos tienen una cantidad de créditos de cursos tomados en el semestre. La probabilidad de que un alumno tenga una cierta cantidad de créditos aparece en el cuadro 1 Esta cantidad influye en la cantidad de horas que el alumno le puede dedicar al curso cada semana. La distribución de cantidad de horas es uniforme y el rango de según la cantidad de créditos aparecen en el cuadro 1. La cantidad de horas que le dedican al curso varía semana a semana.

Número de créditos	Ocurrencia	Rango de horas
40 créditos	0.1	$10 - 25$
50 créditos	0.7	$10 - 15$
55 créditos	0.15	$5 - 15$
60 créditos	0.05	$5 - 10$

Cuadro 1: Probabilidad de cantidad de créditos y cantidad de horas según créditos.

Los alumnos dedican el 70 % del tiempo que tienen disponible para el curso en las tareas (H_t) y el otro 30 % lo dedican a estudiar contenidos (H_s). Estos porcentajes son constantes durante todo el semestre. Estas cantidades se distribuyen de forma uniforme en cada uno de los siete días de la semana. Por ejemplo, un alumno que esta semana puede dedicarle 10 horas al ramo, gastará 7 horas en el desarrollo de la tarea y 3 horas en estudiar la materia. Cada día dedicará una hora a la tarea y 25 minutos a estudiar la materia. Si la actividad es el día jueves, la cantidad de horas de estudio de la materia hasta ese día será de 1 hora y 40 minutos.

4.1.2. Personalidades

Cada alumno siempre tiene un (y sólo un) tipo de personalidad. Cada personalidad afecta de distinta forma las características de cada alumno. Pueden permitir que el alumno aprenda fácilmente unos contenidos más que otros, que avance de distinta forma en las Tareas, o que tenga un mejor desempeño en el Examen. Las personalidades son descritas en el cuadro 2.

4.1.3. Manejo de contenidos S_i

Durante el desarrollo del curso, los alumnos van desarrollando el *manejo de contenidos* que refleja el conocimiento que tienen por cada una de las materias. En las fórmulas, estará representado por S_i donde i es una de las materias. Este conocimiento es determinado por la cantidad de horas por semana que el alumno le dedica al curso. Por otro lado, dadas las horas que el alumno le dedica a estudiar un contenido en específico, se define la nota que este espera en cualquier evaluación que tenga sobre ese contenido. El *manejo de contenidos* se calcula como:

$$S_i = \frac{1}{d_i} \times H_s, \tag{1}$$

Donde d_i es la dificultad de la materia y H_s es la cantidad de horas dedicadas a estudiar la materia hasta ese momento. Las dificultades de cada matería se muestran en el cuadro 3.

Personalidad	Características	Factores que afecta
Eficiente	El alumno intenta siempre mejorar la versión de su programa minimizando el código, tiempo de ejecución y memoria. Tiene un mejor desempeño en el contenido de Funcional y Threading, además de un avance más rápido en el desarrollo de Tareas.	Obtienen un punto sobre la nota en las actividades de funcional y threading. Además, logran un 10 % más de avance en cada área de Tareas.
Artístico	El alumno tiene habilidades artísticas que le permiten producir programas que destacan estéticamente. El aspecto visual para él es lo que tiene más importancia, por lo que tiene un mejor desempeño en Interfaz Gráfica, Webservices y en el área de PEP-8 de las Tareas.	Obtienen un punto sobre la nota en las actividades de Interfaz Gráfica y Webservices. Además, un 20 % extra de avance en el área de PEP-8 de cada Tarea.
Teórico	El alumno sabe mucho sobre programación, pero tiene problemas para desarrollar buenos programas. Tiene un mejor desempeño en Programación Orientada a Objetos (OOP), Metaclases y en el Examen, pero generalmente avanza mas lento en el desarrollo de Tareas.	Obtienen un punto sobre la nota en la actividad de metaclases. También obtienen un punto adicional en la nota del examen. Además, tienen un descuento del 10 % en el avance en cada área de Tareas.

Cuadro 2: Tipos de personalidades y sus características.

Contenidos	Dificultad d_i
Programación orientada a objeto	2
Herencia, composición y agregación	2
Listas, set y diccionario	3
Árbol y grafos	5
Funcional	7
Metaclases	10
Simulación	7
Threading	9
Interfaz Gráfica	1
Bytes y Serialización	6
Networking	6
Webservices	5

Cuadro 3: Dificultad de cada contenido del curso

4.2. Nota Esperada $N_{evaluacion,esperada}$

El cuadro 4 muestra la *nota esperada por el alumno* $N_{evaluacion,esperada}$ de acuerdo a la cantidad de horas de estudio que realizó para las diferentes materias del curso. Cada rango de notas tiene una distribución uniforme. Es importante notar que la *nota esperada por el alumno* es la nota que el alumno cree que debería obtener, pero esto no siempre será igual a la nota que le den los ayudantes. Por ejemplo, si un alumno estudia entre 5 y 7 horas la materia de Metaclases, la nota que espera estará entre un 4.0 y 5.9.

Contenidos \ nota esperada	1.1 y 3.9	4.0 y 5.9	6.0 y 6.9	7.0
Programación orientada a objeto	0 - 2	3 - 4	5 - 6	7
Herencia, composición y agregación	0 - 3	4 - 6	7	8
Listas, set y diccionario	0 - 1	2 - 4	5 - 7	7
Árbol y grafos	0 - 2	2 - 5	6 - 7	8
Funcional	0 - 3	4 - 7	8	9
Metaclases	0 - 4	5 - 7	8 - 9	10
Simulación	0 - 3	4 - 6	7 - 8	9
Threading	0 - 2	2 - 5	6 - 7	8
Interfaz Gráfica	0 - 1	2 - 4	5 - 6	7
Bytes y Serialización	0 - 4	5 - 7	8 - 9	10
Networking	0 - 2	2 - 5	6 - 7	8
Webservices	0 - 3	3 - 7	8	9

Cuadro 4: Nota esperada por el alumno, para cada contenido, según las horas de estudio.

4.2.1. Confianza C

Todos los alumnos parten el curso con un nivel de confianza el cual se distribuye uniformemente entre 2 y 12. Este valor cambia durante el trayecto del curso cada vez que el coordinador publica alguna nota y se rige bajo la siguiente fórmula:

$$C = confianza\_anterior + confianza\_notas$$

En donde $confianza\_anterior$ es la confianza antes de conocer su nota las notas y $confianza\_notas$ es un factor que mide la confianza en relación a la nota esperada y la nota obtenida. Para calcular la confianza cuando se publica la primera nota, $confianza\_anterior$ es el nivel de confianza inicial. Para obtener $confianza\_notas$ se utiliza la siguiente expresión:

$$confianza\_notas = 3x \cdot (N_{a,Final} - N_{a,esperada}) + 5y \cdot (N_{t,Final} - N_{t,Esperada}) + z(N_{c,Final} - N_{c,Esperada}),$$

Donde:

- C = confianza actual
- N_a = nota actividad
- N_t = nota tarea
- N_c = nota control.
- $x = 1$ si hay nota de actividad, 0 en otro caso.
- $y = 1$ si hay nota de tarea, 0 en otro caso.
- $z = 1$ si hay nota de control, 0 en otro caso.

4.2.2. Nivel de programación P

Los alumnos iniciarán el curso con un *nivel de programación* el cual refleja la velocidad con la que avanza el alumno en sus evaluaciones. Este valor parte de forma aleatoria entre 2 y 10.

El nivel de programación de los alumnos va aumentando al pasar las semanas y depende del nivel de programación de la semana anterior y de los eventos que ocurren durante la semana (como ir a hacer consultas a un profesor o ir a una fiesta).

La fórmula está dada por:

$$P_n = \begin{cases} \texttt{random(2,10)} & \text{si } n = 1 \\ 1{,}05 \cdot (1 + v - w) \cdot P_{n-1} & \text{si } n > 1 \end{cases}$$

Donde:

$$v = \begin{cases} 0{,}08 & \text{si el alumno se reune con un profesor} \\ 0 & \text{si no} \end{cases}$$

$$w = \begin{cases} 0{,}15 & \text{si el alumno va a una fiesta} \\ 0 & \text{si no} \end{cases}$$

Nota: v y w se explican en detalle en la secciones 4.3 y 6.1 respectivamente.

4.3. Profesor

Los profesores dictan la cátedra del curso. Durante la clase los profesores entregan *tips* sobre la materia. Cada alumno tendrá una probabilidad de haber escuchado esta información o no. De haber sido escuchada, el *manejo de contenidos* para la evaluación que se está realizando aumentará en un 10 %.

Cada profesor tiene asignado un día para recibir consultas de los alumnos en su oficina. Como máximo puede recibir a 10 personas por semana y no puede ser la misma persona por dos semanas seguidas (para darle la oportunidad a otros alumnos). Los alumnos que sean recibidos los profesores podrán resolver dudas sobre el curso, con lo cual aumentan su *nivel para programar*. Un alumno decide visitar al profesor cuando su promedio es menor o igual a 5.0 . Además, si el promedio es mayor 5.0, cada alumno tiene un 20 % de probabilidad de decidir visitar al profesor. Entre todos los alumnos que deciden ir a la oficina, el profesor elige al azar a las 10 personas de esa semana. Las personas que visiten al profesor aumentarán su *nivel de programación* de esa semana en un 8 %. Un alumno puede ir a visitar únicamente al profesor de su sección.

4.4. Ayudante

Parte fundamental del curso es realizado por los ayudantes. Estas personas son las creadoras de las evaluaciones y quienes las corrigen. Además, definen cual será la *exigencia*, es decir el nivel de *progreso* necesario para obtener un 7. El *progreso* se define como el nivel de avance que el alumno logra en una evaluación. Después de dos semanas desde alguna de estas evaluaciones, los ayudantes le entregarán las notas al coordinador para que él las suba.

4.4.1. Ayudante de tarea

Los ayudantes de tarea se encargan de corregir las tareas del curso. Cada semana en que se suba una tarea los ayudantes tendrán una reunión para definir cuál será el progreso mínimo para tener un 7.0. Al finalizar esta reunión se subirá la tarea.

4.4.2. Ayudante de docencia

Los ayudantes de docencia se encargan de corregir controles, actividades y el examen. Ellos tienen que asistir a las cátedras del curso, donde cada ayudante estará disponible para responder dudas durante el transcurso de la evaluación. A cada sección deben ir 3 ayudantes. Cada ayudante puede resolver dudas 200 veces cada clase. Cada alumno pide ayuda entre 1 a 10 veces durante la clase, con distribución triangular de moda 3, y cuando estos son ayudados aumentará su manejo de contenidos en 1 %. Cabe destacar que un alumno puede ser ayudado más de una vez por los ayudantes durante la clase. El orden en que los alumnos piden ayuda debe ser aleatoio. Es posible que algunos alumnos se queden sin resolver sus dudas durante la clase.

Además de estos eventos, cada semana dos ayudantes de docencia elegidos al azar tendrán que dictar una ayudantía con respecto a la materia de la semana. Al inicio de la simulación, a cada ayudante de docencia se le asignan al azar 3 contenidos del curso que serán los que más domina (un contenido del curso puede ser dominado por más de un ayudante a la vez). Cuando a un ayudante le toca dictar la ayudantía de la materia que se destaca, este dará *tips* adicionales que ayudarán a los alumnos aumentando su *manejo de contenidos* en 10 %. Si no se destaca no existirá este incremento.

4.5. Coordinador Dr. Mavrakis

Como este curso es de una dimensión paralela y el coordinador es el Malvado Dr. Mavrakis, él es capaz de atrasar la subida de notas entre 2 y 5 días, lo que ocurre con un 10 % de probabilidad. También, al momento de subir las notas puede decidir aplicarles o no un descuento de 5 décimas a todos los alumnos con el fin de hacerlos sufrir por un tiempo. Si decide aplicar el descuento, la nota verá reflejado dicho descuento hasta la siguiente publicación de notas donde se le quitará el descuento. Este *descuento arbitrario* puede ocurrir entre 0 y 3 veces por semestre y si pasa una vez no puede volver a pasar por un mes. Este descuento jamás lo aplicará en la última evaluación que suba del semestre. Cada vez que se suben las notas existe un 50 % de probabilidad de que las notas se suban con descuento.

5. Evaluaciones (9 %)

El curso tiene 4 evaluaciones diferentes: controles sorpresa, actividades, tareas y examen. Las tareas y las actividades tienen los siguientes aspectos a evaluar: *PEP-8*, *contenidos* y *funcionalidad*. Los controles y el examen no evalúan *PEP-8*. Los alumnos logran un *progreso* en cada aspecto de la evaluación y un *progreso total*. Las notas se asignan de acuerdo a la *exigencia* que establezcan los ayudantes y profesores.

5.1. Actividades

Las actividades se realizan cada semana y evalúan un contenido distinto, partiendo la semana 1 con Programación Orientada a Objetos hasta la semana 12 con Webservices. Al momento de realizar la actividad, el alumno debe estimar la nota que obtendrá dependiendo de la cantidad de horas que haya podido estudiar, según el cuadro 4.

Cada actividad se evalúa en tres áreas: *PEP-8*, *contenidos* y *funcionalidad*. El nivel de progreso de una actividad estará dado según el *manejo de contenidos* que tenga de esa materia (S_i), el *nivel de programación* (P) y la *confianza* en ese momento (C) según la siguiente fórmula:

$$Progreso_{PEP-8} = 0{,}7 \times S_i + 0{,}2 \times P + 0{,}1 \times C$$

$$Progreso_{funcionalidad} = 0{,}3 \times S_i + 0{,}7 \times P + 0{,}1 \times C$$
$$Progreso_{contenidos} = 0{,}7 \times S_i + 0{,}2 \times P + 0{,}1 \times C$$

5.2. Tareas

Las tareas, como son un trabajo para la casa, poseen una forma de evaluar distinta. El *progreso* de la tarea se define como una ponderación de los *progresos* en cada área. Mientras que el *progreso* en cada *área* depende de las horas que le dedicó el alumno a la tarea (H_t), el manejo de los contenidos (S_i) y su nivel de programación (P).

$$Progreso_{PEP-8} = 0{,}5 \times H_t + 0{,}5 \times P$$
$$Progreso_{contenido} = 0{,}7 \times S_i + 0{,}1 \times P + 0{,}2 \times H_t$$
$$Progreso_{funcionalidad} = 0{,}5 \times S_i + 0{,}1 \times P + 0{,}4 \times H_t$$

En caso de que el de progreso sea menor al 50 % del progreso esperado al momento de entregar, el alumno mandará un mail pidiendo que se dé mas tiempo. Hay un 20 % de probabilidad de que se den dos días mas para una tarea si el 80 % del total de los alumnos mandan un mail pidiendo mas tiempo, pero si es que dan mas tiempo, en ninguna otra tarea se dará mas plazo.

5.3. Controles sorpresa

Al inicio de cada actividad los alumnos pueden tener un control sorpresa. Los alumnos no pueden tener mas de 2 controles seguidos y en total máximo 5 controles. El progreso de un control estará dado por la siguiente fórmula:

$$Progreso_{contenidos} = 0{,}7 \times S_i + 0{,}05 \times P + 0{,}25 \times C$$
$$Progreso_{funcionalidad} = 0{,}3 \times S_i + 0{,}2 \times P + 0{,}5 \times C$$

5.4. Examen

La evaluación final se compone de 8 preguntas, 2 preguntas son de las dos materias que tuvieron mejor promedio como curso y las otras 6 son de la materia que tuvo peor promedio como curso. Para esto tienes que sacar el promedio de notas en el control, actividad y tarea de la materia (si es que hubo) y luego promediar esas 3 notas para tener el promedio final por materia.

El examen se rinde 5 días después de que el coordinador publica todas las notas. Por cada pregunta se obtiene un progreso con la siguiente fórmula.

$$Progreso_{contenido} = 0{,}5 \times S_i + 0{,}1 \times P + 0{,}4 \times C$$
$$Progreso_{funcionalidad} = 0{,}3 \times S_i + 0{,}2 \times P + 0{,}5 \times C$$

5.5. Evaluación

Las actividades y las tareas obtienen su progreso final según la siguiente fórmula:

$$Progreso_{total} = 0{,}4 \times Progreso_{funcionalidad} + 0{,}4 \times Progreso_{contenido} + 0{,}2 \times Progreso_{PEP-8}$$

Cada control y cada pregunta de examen obtiene su progreso final según la siguiente fórmula:

$$Progreso_{total} = 0{,}3 \times Progreso_{funcionalidad} + 0{,}7 \times Progreso_{contenido}$$

El progreso del examen es el promedio de los progresos de todas las preguntas del examen.

Cuando las evaluaciones son creadas, los ayudantes y profesores establecen un nivel de exigencia. Este es el nivel de progreso mínimo para obtener un 7 en la evaluación. La nota de los alumnos se calcula de la siguiente forma:

$$N_{evaluacion,final} = max(\frac{Progreso_{total}}{Exigencia} \times 7, 1)$$

La exigencia para cada evaluación se define como:

$$exigencia = 7 + \frac{\texttt{random(1,5)}}{d_i},$$

donde d_i es la dificultad de la materia de la evaluación (ver cuadro 3).

6. Eventos (19 %)

A continuación se encuentran los eventos de los distintos integrantes del curso que son imprescindibles para realizar la simulación. Pueden haber más pero al menos deben considerar los siguientes:

- Ayudantes: Subir notas (8 a 15 días después de rendir o entregar tarea)
- Alumno: Cátedra (cada 7 días)
- Alumno: Entregar tarea (cada 14 días después de subir la tarea)
- Alumno: Rendir Examen (5 días después de subida la ultima nota)
- Ayudantes Docencia: Reunión para definir dificultad de actividad y cual será el nivel de progreso esperado de la evaluación (cada 7 días).
- Ayudantes Tareas: Reunión para definir dificultad de la tarea y cual será el nivel de progreso esperado de la evaluación (cada 14 días). Al finalizar la reunión se publica la tarea.
- Alumno y ayudante: Ayudantía (cada 7 días)
- Bota de ramo: Al momento de recibir la nota de la cuarta actividad, el alumno analizará sus notas y la confianza que tiene en ese momento del curso dada la siguiente formula.

  $$s = Confianza * 0{,}8 + Notas * 0{,}2$$

  Si s es menor a 20, el alumno decidirá botar el curso.

6.1. Eventos No programados

Durante el transcurso del semestre. los alumnos no solo dedican su tiempo a Avanzación Programada, sino que tienen otros eventos fuera del curso que afectan el rendimiento de este. Cada tipo de evento distribuye `expovariate` de tasa λ. Los tipos de eventos son los siguientes:

- Alumno: Fiesta ($\lambda = \frac{1}{30}$). Cuando sucede una fiesta, van 50 personas en total de forma aleatoria. En este suceso todos los alumnos que asisten pierden un 15 % en el *nivel de programación* de esa semana además de 2 días de estudio.
- Alumno y Ayudante: Partido de fútbol ($\lambda = \frac{1}{70}$). Cuando hay un partido de fútbol, un 80 % de los alumnos ese día no dedicarán tiempo al ramo, y los ayudantes harán que la siguiente tarea necesite de un progreso 20 % más alto.
- Todos: Corte de agua ($\lambda = \frac{1}{21}$). La cantidad de alumnos que recibirán los profesores en sus oficinas para ayudar esa semana bajará a 6. Como máximo hay uno de estos eventos durante la semana.

7. Estadísticas (14 %)

Al ocurrir cada evento, se debe imprimir en consola qué **tipo de evento** ocurrió, ya sea la entrega de una tarea, rendición de un control, que se entregue alguna nota, etc. También se debe mencionar **en qué tiempo** fue y por **cuántas personas** fue realizado, en caso de involucrar una cantidad específica. Algunos ejemplos de cómo se debería mostrar cierta información:

```
>>120 alumnos realizaron el control 3 la semana 7.
>>Se entregaron las notas del control 3 con promedio 4.0 la semana 9.
>>104 alumnos entregaron la tarea 4 la semana 10.
```

Si quiere simplificar esto puede utilizar la librería `logging`.

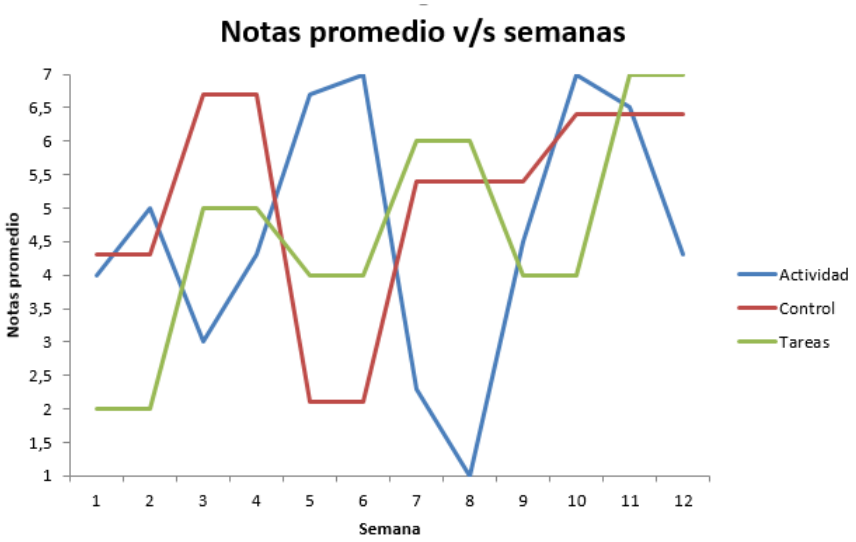
7.1. Gráficos

Al finalizar el programa, se debe crear gráfico que represente las estadísticas a lo largo del semestre.
En el gráfico, el eje x representa el tiempo de la simulación, para representar como varían las notas a medida que pasa el semestre. La unidad de medida de este eje es una semana. El eje y, en cambio, debe indicar la nota. El gráfico debe contener los promedios entre los alumnos de las siguientes variables:

- Nota de cada control
- Nota de cada actividad
- Nota de cada tarea
- Nota del examen

Para estas notas se deben considerar las notas **publicadas** y sin contar a los alumnos que botaron el ramo.

En la imagen 7.1 se muestra un ejemplo de cómo debería ser el gráfico en la semana 12:



7.2. Estadísticas Personales

Al final de la simulación, se debe poder ingresar el nombre de un alumno, para ver toda su información. Se deben dar dos opciones:

- Ver sus cualidades: nivel de programación promedio, confianza y manejo de contenidos
- Notas de cada evaluación y su promedio final

7.3. Estadísticas Finales

Al final del programa se deberá imprimir una lista con la siguiente información:

- Cantidad total de alumnos que quisieron botar el ramo.
- Promedio de confianza al inicio y al final del ramo.
- Mes en que más aprobación tuvo el ramo, ponderando 15 % las actividades, 35 % las tareas y 50 % el examen. Si no hay algún tipo de evaluación en algún mes, asumir que las faltantes son un 1.0. Por ejemplo si solo hubo actividades y tareas, asumir que el examen es un 1.0.
- Tareas, actividades y examen aprobados v/s reprobados. Debe ser entregado en porcentajes. Se debe entregar para cada tarea y actividad, no su promedio.
- Finalmente, debe decir el porcentaje de alumnos que aumentaron su confianza y cuántos la disminuyeron (en cada actividad, tarea, examen).

8. Análisis de resultados (14 %)

Una parte muy importante de simulación es analizar como cambian los resultados obtenidos en distintos escenarios. Junto al enunciado de la tarea se entrega el archivo `escenarios.csv` el cual contiene en cada

columna un conjunto de valores para asignarle a los parámetros que son usados en la simulación. Estos valores deben ser ingresados como input de su simulación. Además, su consola debe incluir una opción para comparar los resultados de todos los distintos escenarios indicando cual es aquel que maximiza el porcentaje de aprobación de los estudiantes.

Si el valor de alguno de los parámetros es un "-", significa que se debe utilizar el valor de ese parámetro del escenario por defecto (descrito en el enunciado). Puedes asumir que las combinaciones de parámetros en un mismo escenario serán consistentes, por ejemplo, las probabilidades de que un alumno tenga 40, 50, 55 o 60 créditos sumará 1. Es importante destacar que cualquier parámetro de la simulación que no se encuentre en escenarios.csv debe ser ingresado con el valor que se indique en el enunciado, en caso de que no se especifique un valor queda a su criterio el valor a utilizar.

Los nombres de los parámetros que aparecer en escenarios.csv son los siguientes:

- `prob_40_creditos`: Probabilidad que los alumnos tengan tomados 40 créditos. (debes considerar este parámetro y los 3 siguientes a la hora de comparar ya que necesitas las otras probabilidades de cuantos créditos tienen los alumnos.)
- `prob_50_creditos`: Probabilidad que los alumnos tengan tomados 50 créditos.
- `prob_55_creditos`: Probabilidad que los alumnos tengan tomados 55 créditos.
- `prob_60_creditos`: Probabilidad que los alumnos tengan tomados 60 créditos.
- `prob_visitar_profesor`: Probabilidad de que un alumno con promedio sobre 5 visite al profesor.
- `prob_atraso_notas_Mavrakis`: Probabilidad de que el malvado Dr. Mavrakis atrase la entrega de notas.
- `porcentaje_progreso_tarea_mail`: Porcentaje de progreso de la tarea que deben tener para no enviar un mail pidiendo más tiempo (si es menor a esta probabilidad el mail se envía).
- `fiesta_mes`: Cantidad de veces que ocurre el evento fiesta en un mes.
- `partido_futbol_mes`: Cantidad de veces que ocurre el evento partido de fútbol en un mes.
- `nivel_inicial_confianza_inferior`: Corresponde al primer parámetro que recibe la uniforme que simula el nivel de confianza de cada alumno al comenzar el curso.
- `nivel_inicial_confianza_superior`: Corresponde al segundo parámetro que recibe la uniforme que simula el nivel de confianza de cada alumno al comenzar el curso.

El cuadro 5 muestra un ejemplo con tres escenarios.

Parametro	escenario_1	escenario_2	escenario_3.
prob_40_creditos	0.2	0.5	0.1
prob_50_creditos	0.4	0.05	0.1
prob_55_creditos	0.25	-	0.1
prob_60_creditos	0.15	0.2	0.7
fiesta_mes	1	3	6
porcentaje_progreso_tarea_mail	50 %	30 %	80 %
prob_visitar_profesor	0.70	0.60	0.50

Cuadro 5: Cuadro Valores Simulación

9. Bases de datos (5 %)

Junto al enunciado de la tarea, se le entregará el archivo `escenarios.csv` (explicado en la sección 8) e `integrantes.csv` con nombres, sus respectivos roles dentro del curso y finalmente a la sección a la que pertenecen. Es importante destacar que los ayudantes no tienen una sección en particular, por lo tanto tendrán ese valor vació dentro del archivo. Además, recibirá el archivo `parámetros.csv` con la cantidad de profesores, la cantidad de ayudantes y una cantidad de alumnos. La tabla de integrantes será como el cuadro 6.

Nombre	Rol	Sección
Mavrakis	Coordinador	
Antonio	Tarea	
Ivania	Profesor	1
Sebastian	Docencia	
Hugo	Alumno	2

Cuadro 6: Cuadro de integrantes curso

10. Requisitos (15 %)

Se evaluará la calidad de su código. En particular, su código debe contener lo siguiente:

- Uso de properties
- No usar referencia circular en la modelación de clases. Revisar esta issue para mayor detalle. Si tiene dudas, pueden crear una issue para mejor orientación sobre la modelación
- Documentar métodos, clases y funciones. Todos sus módulos deben estar documentos en donde se explica cada objeto e indican el tipo de dato de los argumentos recibidos y retornados junto con una explicación de ellos. Sobrescribir un método de la clase object(__str__, __add__) no requiere documentación y en el __init__ no se debe indicar información sobre lo retornado:

```
class Auto:
    """
    Esta clase representa a un Auto...
    """
    def __init__(self, patente):
        """
```

```
            :param patente: Identificador unico para cada auto
            :type patente: str
            """
            self.patente = patente

        def __str__(self):
            return self.patente

    class Estacionamiento:
        """
        Esta clase representa a un estacionamiento capaz de almacenar una gran
        cantidad de autos, tambien permite el retiro de estos...
        """
        def __init__(self, enncargado, autos_dentro):
            """
            :param enncargado: Nombre del enncargado del estacionamiento
            :type enncargado: str
            :param autos_dentro: Todos los autos que el estacionamiento ya posee dentro
            :type autos_dentro: list(Auto)
            """
            self.enncargado = enncargado
            self.autos_dentro = autos_dentro

        def retirar_auto(self, patente):
            """
            Busca un auto según su patente y lo retira del estacionamiento
            :param patente: String que indica la patente del auto
            :type patente: str
            :return: Auto con igual patente a la buscada
            :rtype: Auto
            """
            index = 0
            auto_por_retirar = None
            encontrado = False
            while index < len(self.autos_dentro) and not encontrado:
                if self.autos_dentro[index].patente == patente:
                    auto_por_retirar = self.autos_dentro.pop(index)
                    encontrado = True

            if encontrado:
                return auto_por_retirar
```

Además, deberá entregar un Diagrama de Eventos. Para éste diagrama, se les pide listar todos posibles eventos que pueden ocurrir durante la simulación de la forma:

$\texttt{evento}_1$|cuándo ocurre|qué pasa cuando ocurre el $\texttt{evento}_1$ (cambios, actualizaciones, etc)
$\texttt{evento}_2$|cuándo ocurre|qué pasa cuando ocurre el $\texttt{evento}_2$ (cambios, actualizaciones, etc)

.
.
.

$\texttt{evento}_n$|cuándo ocurre|qué pasa cuando ocurre el $\texttt{evento}_n$ (cambios, actualizaciones, etc)

11. Notas

- Cualquier detalle que no esté mencionado en el enunciado realice supuestos razonables.

- Si considera que la simulación se puede mejorar cambiando alguna constante o alguna formula puedo hacerlo mientras lo especifique en el `README`.
- Es importante destacar que todo lo que sucede en esta simulación es en una dimensión paralela Y NO REPRESENTA AL CURSO DE PROGRAMACIÓN AVANZADA. Cualquier similitud con la realidad es mera coincidencia.

12. Restricciones y alcances

- Desarrolla el programa en Python 3.5.
- La tarea es individual, y está regida por el Código de Honor de la Escuela: Click para Leer.
- Su código debe seguir la guía de estilos `PEP8`.
- Si no se encuentra especificado en el enunciado, asuma que el uso de cualquier librería Python está prohibido. Pregunte en el foro si se pueden usar librerías específicas.
- El ayudante puede descontar el puntaje de hasta 5 décimas de tu tarea, si le parece adecuado. Recomendamos ordenar el código y ser lo más claro y eficiente posible en la creación de los algoritmos.
- Adjunte un archivo `README.md` donde comente sus alcances y el funcionamiento de su sistema (*i.e.* manual de usuario) de forma *concisa* y *clara*. Tiene hasta 24 horas después de la fecha de entrega de la tarea para subir el `README.md` a su repositorio.
- Separe su código de forma inteligente, estructurándola en distintos módulos. Divídalas por las relaciones y los tipos que poseen en común. **Se descontará hasta un punto si se entrega la tarea en un solo módulo.**
- Cualquier detalle no especificado queda a su criterio, siempre que no pase por sobre los requerimiento definidos en el enunciado.

13. Entrega

13.1. Diagrama de eventos

- **Fecha/hora:** jueves 11 de mayo del 2017, 23:59 horas.
- **Lugar:** Se abrirá un cuestionario en SIDING.

13.2. Código

- **Fecha/hora:** viernes 19 de mayo del 2017, 23:59 horas.
- **Lugar:** GIT - Carpeta: Tareas/T04

Tareas que no cumplan con las restricciones señaladas en este enunciado tendrán la calificación mínima (1.0).