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

IIC2233 - Programación Avanzada
2^o semestre 2016

Actividad 08

Manejo de excepciones y Testing

Instrucciones

En un lugar muy lejano, a un costado de Fantasilandia, en la USE (Universidad Sin Estadio), un grupo de alumnos fue seleccionado para programar una simulación de la nueva prueba de selección para la educación superior OnLine.
Es por esto que el gobierno, al ver los exitosos programas que hacen los alumnos de Programación Avanzada de la PUKE, te decidieron contratar a ti, para revisar el programa y encontrar todos los posibles bugs que hacen que el sistema no funcione bien.
Para lograr esto, te pidieron que realices una serie de tests que corroboren que el sistema funcione a la perfección, y además, verifiquen algunos errores usuales que el código presenta.
El Ministerio de Educación te entregó el programa hecho por la USE y las pruebas de matemáticas y lenguaje con sus respectivas preguntas y respuestas.

La Simulación

El código simula distintas situaciones que pueden ocurrir en el transcurso del periodo de pruebas. Entre ellas se encuentran:

1. La creación de las pruebas de matemáticas y lenguaje con un archivo tipo txt. Cada prueba tiene cantidad indefinida de preguntas y cada pregunta solo tiene respuesta A, B y C .
2. El ingreso de respuestas equivocadas (que no son ni A, ni B, ni C) por los alumnos ficticios mientras realizan las pruebas.
3. La búsqueda del alumno que obtuvo el x-lugar en alguna prueba.
4. Alumnos que intentan ver resultados de alguna prueba, independiente de si la dieron o no .
5. Alumnos que se inscriben tarde para alguna prueba por lo que no alcanzan a darla.
6. Entre otras...

Lamentablemente, el equipo informático (Remde) nunca ha sabido manejar de buena manera las contraseñas de sus usuarios (que para esta prueba son los RUT de los alumnos). Es por esto que este año han decidido implementar la autenticación en dos pasos, donde al intentar ver sus resultados se le envía al respectivo alumno una clave de dos dígitos. Esta clave se obtiene aplicando la siguiente formula al rut sin el dígito verificador:

$$(98 - (rut * 100 \mod 97)) \mod 97$$

De ahí se obtienen los dos dígitos verificadores. Luego, si al concatenar el rut y los dígitos verificadores se tiene que

$$(rut + digitos) \mod 97 = 1$$

Entonces se procede a ver los resultados del alumno.

Todo lo anterior ya fue programado por el equipo de la USE, por lo que **tu deber será revisar los distintos errores que pueda presentar el código: debes verificar que no se caiga el sistema**. Además, deberás crear otro `archivo.py` que **verifique que los métodos utilizados en el programa efectivamente estén funcionando correctamente**. En los requerimientos se explicará, con más detalle, los distintos tests a realizar en esta actividad.

Requerimientos

- En esta actividad debes revisar el programa `main.py` y editar el código de tal forma de manejar todas las excepciones que se cometen durante la simulación del desarrollo de las pruebas.
- En el uso de except se debe incluir el tipo de excepción. De otro modo se considerara incorrecto.
- Es importante que **no modifiques la funcionalidad del código entregado**. Tu deber es solo manejar las excepciones, pero sin cambiar el funcionamiento de los métodos en sí.
  - En caso de caer en una situación de excepción, se debe imprimir en pantalla cual fue el origen/causa del error.
- Además deberás hacer los siguientes tests para mostrarle a los de la USE que el código efectivamente funciona bien.
  - *testAlternativaCorrecta* : chequea que la alternativa almacenada en la PrograSU es la misma de el archivo de texto.
  - *testSoloUnaCorrecta* : comprueba que solo hay una alternativa correcta y 2 incorrectas
  - *testCodigoVerificadorRUT* : comprueba que el dígito verificador del rut esta bien calculado.
  - *testClave* : comprueba que cualquier numero cumple con el algoritmo de autenticación utilizado por el Remde.
  - *testCrearUsuario* : comprueba que el usuario es una instancia de la clase *Student* del `main.py`.
  - *testRegistroExitoso* : comprueba que al registrar un numero aleatorio **n** entre 1 y 10 usuarios la lista de alumnos aumenta en n su largo.

To - DO

- (3.00 pts) Manejo de excepciones
- (3.00 pts) Testing: cada test vale 0.5 pts.

Tips

- Recuerden iniciar los nombres de los tests con la palabra *test*, con *t* minúscula.
- Al principio el código debe caerse en la linea 114. Hay que comentar la linea para seguir avanzando.

Entrega

- **Lugar:** GIT - Carpeta: Actividades/AC08
- **Hora:** 16:55