Tecnología de la Programación - Curso 2013/2014
Grado en Ingeniería Informática, Grado en Ingeniería de Computadores
Grado en Ingeniería del Software, Doble grado en Matemáticas-Informática
Examen de junio - Parte práctica - Duración: 2,5 horas.

Instrucciones

- Lee cuidadosamente el examen hasta el final. Algunas decisiones de diseño tomadas para implementar las primeras partes pueden afectar a la complejidad en la implementación de las últimas.
- En el examen deberás partir de la *práctica 5* realizada durante el curso. **En breve se indicará el procedimiento de acceso a dicho código.**
- El código entregado ***debe compilar***.
- En la corrección del examen se valorará el funcionamiento, la claridad del código, el uso conveniente de los medios proporcionados por la Programación Orientada a Objetos (herencia, polimorfismo...) y comentarios. Para la evaluación se tendrá en cuenta tanto lo pedido en el examen como el código del que se parte (implementación de la práctica 5). Romper la encapsulación de las clases (acceso a atributos privados y protegidos desde clases externas, utilización de atributos públicos, etc.) implica suspender el examen.
- Incluye, además del código, una descripción de los pasos y modificaciones que has realizado para implementar lo que se pide en el examen. Este informe de cambios puedes hacerlo en papel o adjuntarlo como un fichero de texto con el resto de los archivos de código fuente. Adjunta este fichero de texto `descripcion.txt` con el resto de los archivos de código fuente. Deja el fichero en "el raíz" del .zip entregado.
- La puntuación de cada apartado está calculada *sobre 10 puntos*, aunque supondrá el 40% de la nota final.

Consejos

Es preferible realizar el examen en el orden siguiente:

1. Implementar cada uno de los apartados. Tras la implementación de cada parte, comprobar que el código compila y funciona y guardar una copia del mismo. De este modo siempre tendremos algo que entregar que sabemos que compila y que funciona (aunque no esté completo).
2. Revisar y corregir los posibles errores o deficiencias que se arrastren de la práctica base, pues ese código también se evalúa.

Instrucciones de entrega

- Para entregar la solución al examen, crea un fichero zip. En él debes incluir todo el proyecto una vez limpiado de archivos intermedios, y con el fichero `alumnos.txt` con tu nombre completo.
- Nombra al fichero `NN_Apellido1Apellido2.zip`, donde `NN` indica el número de grupo (con dos dígitos).
- La estructura del fichero será la misma que la utilizada durante el curso.
- Si optas por escribir la descripción de los pasos y modificaciones realizadas durante el examen en un fichero de texto, inclúyelo en el raíz del archivo comprimido.
- Para entregar el examen, se utilizará el mecanismo de entregas disponible en el laboratorio (icono *EXÁMENES en LABS Entregas y Recogidas*).
- Antes de abandonar el laboratorio debes pasar por el puesto del profesor para asegurarte de que lo que se ve en el puesto del profesor es lo que has entregado y firmar en la hoja de entregas.

Tecnología de la Programación - Curso 2013/2014
Grado en Ingeniería Informática, Grado en Ingeniería de Computadores
Grado en Ingeniería del Software, Doble grado en Matemáticas-Informática
Examen de septiembre - Parte práctica - Duración: 2 horas.

Enunciado

Vamos a ampliar la funcionalidad de la **Práctica 5** incluyendo un banco de flags a nuestra CPU. Un flag es una señal que se activa después de cada ejecución de una instrucción y que representa si una determinada condición se cumple tras la ejecución de una instrucción. En particular vamos a tener 2 flags distintos: NEG y PAR para indicar si el resultado de la ejecución de una instrucción ha producido un valor negativo o un valor par, respectivamente. Todos los flags se actualizarán tras la ejecución correcta de cualquier instrucción aritmética.

Así mismo vamos a crear instrucciones de salto *indirecto* que hacen uso de estos flags para generar saltos de acuerdo al valor contenido en los flags. Un salto indirecto es el que hace uso del contenido de la memoria para saber a qué dirección ha de saltar.

Por último modificaremos la interfaz de Swing para que se pueda mostrar el estado del banco de registros.

En todos los apartados (salvo el de Swing) se valorará especialmente la extensibilidad del código, de modo que pudieran incluirse en el futuro nuevos flags con el menor esfuerzo posible.

[2 puntos] Parte A

Crea una clase `BancoFlags` con todos los métodos necesarios para poder interactuar con ella. Inicialmente todos los flags tienen un valor que indica que la condición que representan no se cumple.

[2 puntos] Parte B

Integra el banco de flags con la CPU de modo que se actualicen después de la ejecución de cada una de las instrucciones aritméticas ya creadas: ADD, SUB, MUL, DIV. Recuerda que los flags no deben ser actualizados si la instrucción no se ejecuta correctamente.

[1,5 puntos] Parte C

Haz que en el modo Interactivo podamos ver el estado del banco de flags junto con el estado de la CPU. A modo de ejemplo, tras ejecutar el siguiente programa:

```
0: PUSH 5
1: PUSH -1
2: ADD
3: PUSH 7
4: SUB
5: HALT
```

mostramos el estado de la CPU tras ejecutar cada una de las instrucciones (el `*` indica que el flag está activado):

```
Comienza la ejecución de PUSH 5
El estado de la máquina tras ejecutar la instrucción es:
Memoria:  <vacia>
Pila de Operandos: 5
Flags :PAR [ ] NEG [ ]
Comienza la ejecución de PUSH -1
El estado de la máquina tras ejecutar la instrucción es:
```

```
Memoria:  <vacia>
Pila de Operandos: 5 -1
Flags :PAR [ ] NEG [ ]
Comienza la ejecución de ADD
El estado de la máquina tras ejecutar la instrucción es:
Memoria:  <vacia>
Pila de Operandos: 4
Flags :PAR [*] NEG [ ]
Comienza la ejecución de PUSH 7
El estado de la máquina tras ejecutar la instrucción es:
Memoria:  <vacia>
Pila de Operandos: 4 7
Flags :PAR [*] NEG [ ]
Comienza la ejecución de SUB
El estado de la máquina tras ejecutar la instrucción es:
Memoria:  <vacia>
Pila de Operandos: -3
Flags :PAR [ ] NEG [*]
```

[2 punto] Parte D

Implementa las instrucciones de salto con indirección `JUMPNIND n` y `JUMPPIND n` que provocan un salto indirecto en caso de que el flag NEG o PAR esté activado, respectivamente. En caso de que el correspondiente flag esté activado, cambiará el PC con el valor contenido en la posición `n` de la memoria. La ejecución de la instrucción ha de fallar si la dirección de memoria está vacía.

[2,5 puntos] Parte E

Haz las modificaciones necesarias para hacer que el banco de flags se muestre en la interfaz de Swing. Como se puede ver en la Figura, el banco de flags aparece en un panel en la parte superior de la interfaz. Como se puede ver, el flag de paridad se ha activado tras la ejecución de la instrucción `ADD` (8 es par pero no es negativo).

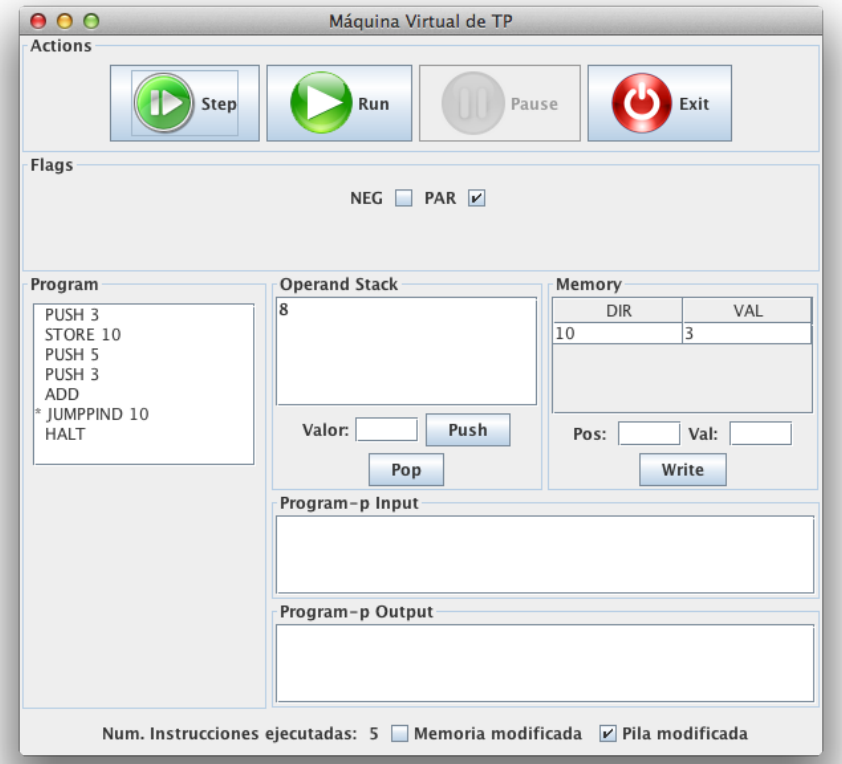

Figure 1: El aspecto de la interfaz