UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
CAMPUS SANTIAGO

LENGUAJES DE PROGRAMACIÓN
Certamen Final (2)

Profesor: José Luis Martí L.
Ayudantes: Sebastián Campos M., Gabriel Carmona T., Sebastián Godínez G.

Parte 1: Verdadero o Falso (2 puntos cada una)

1. Los lenguajes imperativos están dominados por asignaciones a variables que modifican el estado del programa, con potenciales efectos laterales. (V)
2. Las variables a veces son leídas de la memoria y otras, están incrustadas en las instrucciones de máquina. (F)
3. Un ejemplo de efecto lateral son las referencias a variables no locales. (V)
4. La técnica por expansión convierte un objeto a un tipo que no puede incluir todos los valores del tipo original. (F)
5. Tanto una expresión relacional como una *booleana* entregan un resultado *booleano*, cuando el lenguaje de programación tiene este tipo de datos. (V)
6. Sólo el lenguaje de programación C considera a la asignación como un operador, con la posibilidad de incrustarlo en cualquier expresión. (F)
7. Aquellos ciclos controlados por estructuras de datos son casos típicos de ciclos en C, Python y Ruby. (F)
8. Las interfaces de los subprogramas pueden incluir elementos como nombre, parámetros y excepciones. (V)
9. La sobrecarga de subprogramas surge con lenguajes de programación como C, C++ y C#.
10. El concepto de ocultamiento de información considera que la declaración del tipo y los protocolos de las operaciones sobre los objetos del tipo están contenidos en una única unidad sintáctica. (F)
11. Al incorporar la palabra reservada “private” en una clase Python, se asegura que los atributos definidos en su interior sólo puedan ser accedidos dentro de la misma. (F)
12. Una máquina virtual realiza la carga de clases que se necesitan en medio de la ejecución, de manera dinámica. (V)
13. Una clase puede extender varias interfaces, pero sólo implementar una única superclase. (F)
14. C++ es un lenguaje orientado al objeto, que no soporta programación procedural. (F)
15. Una clase abstracta tiene todos sus métodos también abstractos. (F)
16. Para que una variable miembro pueda ser compartida por todos los objetos de una misma clase, se debe usar el modificador static. (V)
17. Con la herencia se tiene que considerar que a una variable de un tipo base se le pueden asignar referencias de cualquier tipo derivado. (V)
18. En una jerarquía de clases donde se utilizan métodos redefinidos, de forma dinámica se realiza el ligado al método correspondiente. (V)
19. En la asignación de memoria para la herencia, es preferible que sea el área de stack la utilizada para la creación explícita de objetos. (F)
20. Con el operador new, el runtime asigna memoria suficiente para el objeto que se está creando, además de inicializarlo con algún constructor. (V)
21. Una clase sólo puede tener un único constructor. (F)
22. La referencia this se refiere al objeto sobre el cual se invocó el método. (V)
23. Si el *runtime* no puede asignar memoria suficiente para crear un objeto, se lanza una excepción *OutOfMemoryError* la que llevará a la ejecución del recolector de basura. (F)

24. Un constructor, como todo método, puede ser sobrecargado, es decir una clase puede tener varios constructores con distintas firmas entre sí. (F).
25. Un constructor por defecto es público si la clase a la cual pertenece también lo es. (V)
26. En Java, todos los objetos son asignados desde el heap y accedidos mediante variables de referencia. (V)

Parte 2: Alternativas (8 puntos cada una)

1. ¿Cuál de los siguientes pares de funciones en C++ son un ejemplo de sobrecarga de funciones?:

   a) int **f**(int a, int b) { … }
      int **f**(int x, int y) { … }

   b) int **f**(int a, int b) { … }
      int **f**(int a, int b, int c) { … }

   c) int **f**(int a, int b) { … }
      int **g**(int a, int b) { … }

   d) int **f**(int a, int b) { … }
      int **g**(int *a, int *b) { … }

2. Dado el siguiente código en C:

int x;
 	
void f(int a)
{
  a = 2;
  x++;
}
	
	void main()
	{
	  x = 1;
	  f(x);
	  printf("%d", x);
	}

   ¿Cuál sería el resultado del *printf()* si el parámetro hubiera sido pasado por valor-resultado y referencia, respectivamente?

   a) Por valor-resultado: 2, Por referencia: 1
   b) Por valor-resultado: 3, Por referencia: 3
   c) Por valor-resultado: 2, Por referencia: 2
   d) Por valor-resultado: 2, Por referencia: 3

3. ¿Cuál de los siguientes lenguajes de programación no da soporte completo a los TAD?:
   a) C++
   b) C
   c) Java
   d) Python

4. ¿Cuál de todos los siguientes códigos Python es el único válido como constructor de un TAD *Stack*?:

   a)
```
def __init__(self, max):
    self.maximo = max
    self.stack = []
```

   b)
```
def __init__(max):
    maximo = max
    stack = []
```

   c)
```
def Stack(max):
    self.maximo = max
    self.stack = []
```

   d)
```
def init(self, max):
    self.maximo = max
    self.stack = []
```

JLML/jlml.
160621.