Examen Parcial

1) **Describir como mínimo 3 paradigmas de programación y dar sus características.**

Paradigma funcional
- La computación se realiza mediante la evaluación de expresiones
- Definición de funciones
- Funciones como datos primitivos
- Valores sin efectos laterales, no existe la asignación
- Programación declarativa
- Lenguajes: LISP, Scheme, Haskell

Paradigma imperativo
- Definición de procedimientos
- Definición de tipos de datos
- Chequeo de tipos en tiempo de compilación
- Cambio de estado de variables
- Pasos de ejecución de un proceso

Paradigma orientado a objetos
- Definición de clases y herencia
- Objetos como abstracción de datos y procedimientos
- Polimorfismo y chequeo de tipos en tiempo de ejecución
- Ejemplo en Java

2) **Cuál es la diferencia entre i++ y ++i, dar un ejemplo en código.**

Ambos se diferencian por el valor que retornan (el valor que nos muestran). Por ejemplo:

i++: el valor de retorno es la variable i y luego de eso se ejecuta la operación.
++i: en este caso primero se hace la operación y luego se aplica la variable por lo que los "valores retornados" son diferentes.

3) **Completar el cuadro con la información de PRIORIDAD, siendo 1 más prioritario que 5.**

OPERADOR	PRIORIDAD	OPERADOR	PRIORIDAD
/	4	(expr)	1
--var	3	+expr	3
*	4	+	5
%	4	-	5
Var--	2	&&	5

4) Responder las siguientes preguntas y dar ejemplos:

a. ¿Qué significa un casting en programación?

El casting es un procedimiento para transformar una variable primitiva de un tipo a otro. También se utiliza para transformar un objeto de una clase a otra clase siempre y cuando haya una relación de herencia entre ambas.

```
int num1 = 100;
long num2 = num1;       // Un int cabe en un long

long num2 = 100;        // 100 en un int
```

b. ¿Qué es una función y un procedimiento?

Una **función**, desde el punto de vista de la programación, se define como un proceso que recibe valores de entrada (llamados parámetros) y el cual retorna un valor resultado. Adicionalmente, las funciones son subprogramas dentro de un programa, que se pueden ejecutar desde cualquier parte del programa, es decir, desde otra función, desde la misma función o desde el programa principal, cuantas veces sea necesario.

```
funcion <nombre> ( param1 : tipo1 , ..., paramn : tipon ) : tipo
variables
    <declaraciones>
inicio
    <instrucciones>
    retornar <expresión>
fin_funcion
```

Donde,

- <nombre>: representa el nombre de la función
- $param_i$: representa el parámetro i-ésimo de la función.
- $tipo_i$: representa el tipo del i-ésimo parámetro de la función.

Los **procedimientos** se usan para evitar duplicación de código y conseguir programas
más cortos. Son también una herramienta conceptual para dividir un problema en subproblemas logrando de esta forma escribir más fácilmente programas grandes y complejos.

En el seudolenguaje un procedimiento se define de la siguiente manera

```
procedimiento <nombre> ( param1: tipo1, ..., paramn : tipon)
variables
    <declaraciones>
inicio
    <instrucciones>
fin_procedimiento
```

Donde:

- <nombre>: representa el nombre del procedimiento.
- $param_i$: representa el parámetro i-ésimo del procedimiento.
- $tipo_i$: representa el tipo del i-ésimo parámetro del procedimiento.
- <declaraciones>: representa el conjunto de variables definidas para el procedimiento (diferentes a los parámetros).
- **<instrucciones>**: representa el conjunto de instrucciones que realiza el procedimiento.

c. ¿Qué quiere decir sobrecarga de operadores?

La sobrecarga se refiere a la posibilidad de tener dos o más funciones con el mismo nombre pero funcionalidad diferente. Es decir, dos o más funciones con el mismo nombre realizan acciones diferentes. El compilador usará una u otra dependiendo de los parámetros usados.

```
public class Articulo {
    private float precio;
    public void setPrecio() {
        precio = 0;
    }
    public void setPrecio(float nuevoPrecio) {
        precio = nuevoPrecio;
    }
}
```