Trabajo Practico № 1
Paradigmas de Programación

Dr. Pablo Javier Vidal Unidad 1

Ejercicio 1.

Responder las siguientes preguntas

1. El breve estudio histórico presentado en clase mediante una figura de evolución de lenguajes, no menciona todos los principales lenguajes de programación existentes hoy en día (solo los que han sido particularmente influyentes, en opinión de los autores seleccionados). Si se ha omitido uno de sus lenguajes favoritos o de uso cotidiano, explique por qué cree que es lo suficientemente importante como para incluirlo y muestre dónde encaja su lenguaje seleccionado en dicha figura.
2. Fortran y Cobol son dos lenguajes de hace muchas décadas, muy viejos en cuanto a transparencia al cliente y al programador. ¿Puede indicar porque aún siguen siendo utilizados? ¿Qué características presentan los dos lenguajes teniendo en cuenta lo visto en clase?.
3. ¿Qué estructura presenta Assembler con respecto a las características que definen a un lenguaje (nivel del lenguaje, sistema de tipos, ligaduras, etc).
4. ¿Qué características presenta el lenguaje Python con respecto a su nivel del lenguaje, sistema de tipos, ligaduras, entre otras.
5. ¿Python es compilado o interpretado? Describir el proceso que realiza python desde el momento que se toma el código fuente hasta lograr ser ejecutado en la maquina.

Ejercicio 2.

Indicar cuál de las siguientes afirmaciones referentes a la Programación Declarativa (PD) e Imperativa (PI) es falsa o verdadera. Justifique la respuesta.

1. Las instrucciones de un programa declarativo son un conjunto de inferencias lógicas.
2. Un programa en PD se corresponde con las instrucciones para resolver un problema.

Ejercicio 3.

Distinga entre:

1. Tipos, Ligadura estática de tipos y ligadura dinámica de tipos.
2. Coacción, error de tipos y comprobación de tipos.
3. Compatibilidad de tipos nominal y compatibilidad de tipos estructural.

Ejercicio 4.

En tiempo de ejecución, cuando entramos en un bloque las variables locales ya no contienen los valores que almacenaban la última vez que fue ejecutado dicho bloque. ¿Por qué ocurre esto?

Ejercicio 5.

Dado el siguiente esqueleto de programa escrito en C, dibuja un rectángulo alrededor de cada uno de sus bloques, etiquétalos con las letras A, B, C, etc. y responde a las siguientes preguntas.

```
int x, y, z;

fun1()
{
  int j, k, l;
  {
    int m, n, x;
     ...
  }
}

fun3()
{
  int y, z, k;
   ...
}

void main()
{ ... }
```

1. Mencione un bloque que esté anidado dentro de otro bloque.
2. ¿En qué ámbitos son accesibles a la vez las variables globales x, y, z?
3. ¿Qué variables son accesibles desde la función *main*?
4. ¿Son k de $fun1$ y k de $fun3$ la misma variable? Razona la respuesta.
5. ¿Son las variables globales y y z accesibles desde $fun3$?. Justifique la respuesta.
6. ¿En qué bloques j hace referencia a una variable local? ¿En qué bloques j es una variable no local?
7. ¿En que bloque o bloques pueden usarse a la vez m y k?

Ejercicio 6.

Indica qué valor de x se imprime en el procedimiento sub1 suponiendo tanto ámbito estático como ámbito dinámico.

```
program ej_7;
        var x: integer;

        procedure sub1;
                begin
                        writeln('x = '+, x);
                end;
        procedure sub2;
                var x: integer;
                begin
                        x := 10;
```

```
                    sub1
                end;

begin {principal de ej_7}
        x := 5;
        sub2;
end. {de ej_7}
```

Ejercicio 7.

Indicar cuál de las siguientes afirmaciones es cierta. Justificar su respuesta.

1. Un analizador léxico (scanner) es un programa que divide una secuencia de caracteres (el programa) en una secuencia de componentes sintácticos primitivos: las instrucciones.
2. La compatibilidad de tipos, el alcance de las variables y la signatura de las funciones (coincidencia del número de parámetros en una llamada con los parámetros formales) forman parte de la semántica estática del lenguaje.

Ejercicio 8.

Indicar cuál de las siguientes afirmaciones es cierta. Justificar su respuesta.

1. Una desventaja de los lenguajes de programación declarativos es que, para resolver un mismo problema, se requieren en general más líneas de código que usando otros lenguajes de programación más convencionales, como Python.
2. Al ser de mayor nivel, los programas declarativos resultan más fáciles de mantener que los correspondientes programas imperativos.
3. Los lenguajes declarativos carecen de aplicaciones prácticas.

Ejercicio 9.

Indica cuál de las siguientes afirmaciones sobre la programación declarativa en comparación con la programación imperativa es VERDADERA o FALSA:

1. Mayor potencia expresiva.
2. Mantenimiento más simple
3. Menor tamaño del código producido.
4. Establecer el cómo proceder

Ejercicio 10.

Indica cuál de las siguientes afirmaciones es VERDADERA o FALSA.

El lenguaje ensamblador:

1. Más cerca del lenguaje máquina que de los lenguajes de alto nivel
2. Más cerca de los lenguajes de alto nivel que del lenguaje máquina
3. No es un lenguaje de programación

Ejercicio 11.

Clasifique los siguientes lenguajes para saber si son compilados/intepretados/hibridos.
Lenguajes: Fortran, JavaScript, Mathematica, C,Java, Perl, Pascal, Haskell, Scala, PHP,

Ejercicio 12.

Indica cuál de las siguientes afirmaciones es VERDADERA o FALSA.

En un lenguaje débilmente tipado:

1. Un valor de un tipo puede ser tratado como de otro tipo
2. Un valor de un tipo nunca puede ser tratado como de otro tipo
3. Un valor de un tipo puede ser tratado como de otro tipo siempre que se realice una conversión de forma explícita
4. Las anteriores respuestas no son correctas

Ejercicio 13.

Imperativo, declarativo y orientado a objetos son:

1. Modos de compilar el código fuente de un programa de ordenador
2. Modos de definir el pseudocódigo de un programa de ordenador
3. Paradigmas de programación
4. Las anteriores respuestas no son correctas

Ejercicio 14.

En base a lo aprendido sobre ligaduras, respondas las siguientes preguntas:

1. ¿Qué es una ligadura? ¿Cómo se pueden clasificar los tiempos de ligadura? ¿A qué características o criterios de evaluación, en general, afecta el tiempo de una ligadura?
2. El concepto de ligadura, está vinculado a la sintaxis o a la semántica de un lenguaje de programación? Justifique y ejemplifique.
3. ¿Cuál es la diferencia entre una ligadura implícita y una explicita? Brindar ejemplos de cada una de ellas.

Ejercicios Prácticos

Ejercicio 15.

Implementar los siguientes enunciados en Python utilizando los conceptos básicos de programación de recursividad, sin el uso de módulos extras.

1. Función que devuelva el n-ésimo número de la sucesión de Fibonacci. Que valor devuelve para n=10? y para n=10000?
2. La función potencia(b,n), devuelve el valor de elevar b a la potencia n.

3. Escribir una función recursiva que reciba como parámetros dos cadena de caracteres a y b, una lista X vacía y el índice de inicio. La función devuelve en X una lista con las posiciones en donde se encuentra b dentro de a. Ejemplo:

```
>>> posiciones_de("Un tete a tete con Tete", "te",[],0)
[3, 5, 10, 12, 21]
```

4. Escribir una función booleana recursiva llamada *VectoresIguales* que reciba dos listas como parámetros y devuelva *TRUE* si son iguales (mismos elementos en el mismo orden), o *FALSE* en caso contrario.

5. Desarrollar una función que permita recursivamente armar el la n-esima linea del triangulo de pascal

   Ejemplo:

```
>>> pascal(5)
[1, 4, 6, 4, 1]
```

6. Escriba una función recursiva llamada Invertir que, dada una lista L, la invierta.

7. En una empresa de entrega de correo los trabajos no se hacen como en un bucle iterativo donde todo se va entregando secuencialmente. Para ello existen dos reglas básicas.

   - Si el número de paquetes es mayor a 1 estamos ante la presencia de un administrados el cual puede nombrar dos trabajadores y dividir su trabajo entre ellos
   - Si el número de paquetes es mayor a 1 él es un trabajador y tiene que entregar el paquete a la casa que le ha sido asignada.

   Se pide al alumno desarrollar una función recursiva que ante una lista de posibles casas donde entregar, permita informar la entrega del paquete cuando lo tenga el trabajador. Ejemplo

```
>>> las_casas = ["Casa de Pablo", "Casa de Caro",
>>> "Casa de Caty", "Casa de Luna"]
>>> entrega_de_paquetes(las_casas)
Entrego el paquete en  Casa de Pablo
Entrego el paquete en  Casa de Caro
Entrego el paquete en  Casa de Caty
Entrego el paquete en  Casa de Luna
```

8. Escribir una función que simule el siguiente experimento: Se tiene una rata en una jaula con 3 caminos, entre los cuales elige al azar (cada uno tiene la misma probabilidad), si elige el 1 luego de 3 minutos vuelve a la jaula, si elige el 2 luego de 5 minutos vuelve a la jaula, en el caso de elegir el 3 luego de 7 minutos sale de la jaula. La rata no aprende, siempre elige entre los 3 caminos con la misma probabilidad, pero quiere su libertad, por lo que recorrerá los caminos hasta salir de la jaula.

   La función debe devolver el tiempo que tarda la rata en salir de la jaula. Ejemplo:

```
>>> ratatuille(randint(1,3))
Va por camino 1
Va por camino 1
Va por camino 2
Va por camino 2
Va por camino 3
Tiempo total:  23
```

9. En la vereda de una calle con adoquines unos niños juegan al tejo. Para esto numeran los adoquines de la siguiente forma:

   Los movimientos permitidos del juego son:

   - Al principio del juego los niños se ubican en el adoquín 1.
   - De un adoquín numerado i se puede saltar al adoquín numerado $i + 1$.
   - De un adoquín numerado i se puede saltar al adoquín numerado $i + 2$ (sin pasar por el adoquín $i + 1$). Por ejemplo, el número de caminos posibles para n=3 es 3 y son los siguientes: (0,1,2,3), (0,2,3) y (0,1,3).

   Escriban una función recursiva llamada $Allways(n)$ que calcule el número de caminos posibles para alcanzar un adoquín objetivo numerado con n (mayor que cero).