


# Redictado Taller de Programación 2020

## CLASE 0

### Programación con Pascal

A laptop is placed on a light-colored wooden desk. The laptop screen displays a Pascal program. The background is a plain, light-colored wall.

```
Program HolaMundo;  
Begin  
  WriteLn('Hola mundo');  
end.
```

# Temas de la clase



1. Introducción a la Programación con Pascal.
2. Ejercitación con operaciones de Vector y Lista



# Actividades en Máquina

## ACTIVIDAD 1

a. Edite, Compile y ejecute el programa **ProgramaNumAleatorio.pas**

```
program NumAleatorio;  
  
var num: integer;  
  
begin  
    Randomize;  
    num := random (100); {valores en el intervalo 0 a 99}  
    writeln ('El numero aleatorio generado es: ', num);  
    readln;  
end.
```

b. ¿Qué hace el programa?



# Actividades en Máquina

## ACTIVIDAD 2

Se propone simular un juego que consiste en contar la cantidad de veces que es necesario generar un número al azar (entre 0 a 10) para que coincida con un valor ingresado por teclado.

Modifique el programa de la Actividad 1 para que informe la cantidad de veces necesarias para que se dé coincidencia.



# Actividades en Máquina

## ACTIVIDAD 3: Crear **programaLISTA**

1. Crear un programa que contenga:

```
Program programaLista;  
  
Type  
  elemento=record  
    cod:integer;  
  
  end;  
  lista=^nodo;  
  nodo=record  
    datos: elemento;  
    sig: lista;  
  
  end;  
  
Begin  
End.
```

2. Guardar como **programaLISTA.pas**



# Actividades en Máquina

3. Implemente el módulo **CrearListaAgregarAdelante** que genere valores aleatorios y los agregue *adelante* de la lista recibida por parámetro. Los valores se generan usando la función Random hasta que llega el valor 15. Se sugiere usar Random entre 0 y 15. El módulo debe contener el **procedure AgregarAdelante** que se muestra a continuación.

```
Procedure AgregarAdelante(var L:lista; elem:elemento);  
Var nue:Lista;  
Begin  
    New(nue);  
    nue^.datos:=elem;  
    nue^.sig:=L;  
    L:=nue;  
End;
```

4. Implementar el módulo **ImprimirLista**

5. Invocar en el programa principal a ambos módulos para crear una lista e imprimirla. Compilar y ejecutar.



# Actividades en Máquina

6. Implementar el módulo **CrearListaAgregarAtrás** que genere valores aleatorios y los agregue *atrás* de la lista recibida por parámetro. Los valores se generan usando la función Random hasta que llega el valor 15. Se sugiere usar Random entre 0 y 15. El módulo debe contener el **procedure AgregarAtras** que se muestra a continuación.

```
procedure AgregarAtras (var pri, ult: lista; elem: elemento);
var  nue : lista;
begin
  new (nue);
  nue^.datos:= elem;
  nue^.sig := NIL;
  if pri <> Nil then
    ult^.sig := nue
  else
    pri := nue;
  ult := nue;
end;
```

7. Invocar en el programa principal al módulo **CrearListaAgregarAtras** para generar una nueva lista y luego muestre la lista generada (usar módulo **ImprimirLista**)



# Actividades en Máquina

8. Implementar el módulo **EliminarElemento** para eliminar un valor de una lista de enteros (sin orden) recibida. El módulo debe leer un valor, invocar al procedure **BorrarElemento** e informar el resultado.

```
Procedure BorrarElemento (var pri:lista; valor:integer; var exito: boolean);
var ant, act: lista;
begin
    exito := false;
    act := pri;
    while (act <> NIL) and (act^.datos.cod <> valor) do begin
        ant := act;
        act := act^.sig
    end;
    if (act <> NIL) then begin
        exito := true;
        if (act = pri) then pri := act^.sig
        else ant^.sig:= act^.sig;
        dispose (act);
    end;
end;
```

9. Invocar en el programa principal al módulo **EliminarElemento** y luego mostrar la lista resultante (usar módulo **ImprimirLista**)





# Actividades en Máquina

**10.** Implementar el módulo **CrearListaOrdenada** que genere valores aleatorios y los inserte *en orden* en la lista recibida por parámetro. El módulo debe contener el **procedure InsertarEnLista** que se muestra a continuación.

```
Procedure InsertarEnLista ( var pri: lista; elem: elemento);  
var ant, nue, act: lista;  
begin  
  new (nue);  
  nue^.datos := elem;  
  act := pri;  
  while (act<>NIL) and (act^.datos.cod < elem.cod) do begin  
    ant := act;  
    act := act^.sig ;  
  end;  
  if (act = pri) then pri := nue  
    else ant^.sig := nue;  
  nue^.sig := act ;  
end;
```

**11.** Invocar en el programa principal al módulo **CrearListaOrdenada** para generar una nueva lista y luego mostrarla en pantalla.



# Actividades en Máquina

**12.** Implementar el módulo **EliminarElementoConOrden**, para eliminar un valor de una lista de enteros ordenada recibida. El módulo debe leer un valor, invocar al procedure **BorrarElementoEnOrden** e informar el resultado.  
El módulo **BorrarElementoEnOrden** debe seguir la especificación:

```
Procedure BorrarElementoEnOrden (var pri:lista;  
valor:integer; var exito: boolean);  
  
begin  
  
end;
```

**13.** Invocar en el programa principal al módulo **EliminarElementoConOrden** con la lista ordenada generada con anterioridad y mostrar la lista resultante.



# Actividades en Máquina

## ACTIVIDAD 4: Crear **programaVECTOR.pas**

1. Crear un programa que contenga:

```
Program programaVector;  
  
Const  
  DimF = 10;  
Type  
  vector = Array [ 1..DimF] of integer;  
  
Begin  
  
End.
```

2. Guardar como **programaVECTOR.pas**



# Actividades en Máquina

3. Implementar el módulo **CrearVector** que cargue un vector de enteros (parámetro) con valores aleatorios generados utilizando la función Random hasta que llega el valor 15. El módulo debe usar el **procedure Agregar** que se presenta a continuación:

```
Procedure Agregar (var v: vector; var dimL:integer;
                  elem: integer; var exito : boolean);
Begin
  If (dimL < dimF) then begin
    dimL:= dimL+1;
    v [dimL]:= elem;
    exito := true
  end
  else exito := false;
end;
```

4. Implementar el módulo **ImprimirVector** que reciba un vector y su dimensión lógica y muestre los elementos del vector de la siguiente forma:

```
Vros almacenados:
```

```
-----
19 | 07 | 42 | 76 | 96 | 26 | 19 | 70 |
-----
```



# Actividades en Máquina

5. Invocar en el programa principal a los módulos **CrearVector** e **ImprimirVector** ya implementados.

6. Implementar el módulo **EliminarElemento** para eliminar un valor de un vector de enteros (parámetro) sin orden. El módulo debe leer un valor, invocar al módulo **BorrarElemento** e informar el resultado (si pudo eliminar o no).

```
Procedure BorrarElemento (var v:vector; var dimL:integer; valor:
integer; var exito: boolean);
begin

end;
```

7. Invocar en el programa principal al módulo **EliminarElemento** con el vector creado con anterioridad y mostrar el vector resultante.



# Actividades en Máquina

## **ACTIVIDAD 5. Resolver**

Implementar un programa que procese la información de los participantes de un concurso de preguntas y respuestas. De cada participante se lee el código de participante y su edad. El ingreso de los participantes finaliza cuando se lee el código -1 (que no debe procesarse). El programa debe informar los códigos de los participantes cuya edad supera el promedio de edades.

Adaptar los módulos ya implementados para utilizarlos en este programa.



# Actividades en Máquina

## ACTIVIDAD 6: Resolver

Una empresa desarrolladora de juegos para teléfonos celulares con Android dispone en papel de información de todos los dispositivos que poseen sus juegos instalados. De cada dispositivo se conoce la versión de Android instalada, el tamaño de la pantalla (en pulgadas) y la cantidad de memoria RAM que posee (medida en GB).

Realizar un programa que:

a. Lea cada dispositivo y lo guarde en una lista ordenada por versión de Android. La lectura finaliza con tamaño de la pantalla en 0.

b. Procese la lista para informar:

- La cantidad de dispositivos para cada versión de Android
- La cantidad de dispositivos con más de 3 GB de memoria y pantallas de a lo sumo a 5 pulgadas
- El tamaño promedio de las pantallas de todos los dispositivos.



# Actividades en Máquina

## ACTIVIDAD 7: Resolver

La Facultad de Informática debe seleccionar los 10 egresados con mejor promedio a los que la UNLP les entregará el premio Joaquín V. González. De cada egresado se conoce su número de alumno, apellido, y el promedio obtenido durante toda su carrera.

Implementar un programa que:

- Lea la información de todos los egresados, hasta ingresar el código 0, el cual no debe procesarse.
- Una vez ingresada la información de los egresados, se debe informar el apellido y número de alumno de los 10 egresados que recibirán el premio. La información debe imprimirse ordenada según el promedio del egresado (de mayor a menor).