**TEMA 5**

**HISTORIAS DE USUARIO**

Material:

* 2 2.1-3.Gestion\_de\_los\_proyectos\_de\_desarrollo.ppt

Principales preguntas a responder:

\*Who

\*What

\*Why

Para que hacer historias de usuarios:

Obtener los requerimientos que se requieren en el sistema.

Normalización de bases de datos(5 Reglas)

\*Campo atómico

\*Integridad referencial

\*Seguridad de datos…

**TEMA 6**

**SUBVERSION, GIT Y GITHUB**

* Documentación
* Comandos
* Repaso

**TEMA 7**

**ALGORITMOS**

**Características**

* Finito
* Definido
* Preciso

**Símbolos y operadores son 3 tipos:**

* Aritméticos (+ ,-, \*, =, sqr, sqrt, %, /, pow )
* Comparación ( <, >, <=, >=, ==, ¡=)
* Lógico ( &&, ||, ¡)

Se puede representar en Pseudocódigo (lenguaje natural) y en diagramas de flujo (notación estándar).

Siempre al representar **Seudocódigo** se inicia con INICIO y se termina con FIN.

Las variables es reservar espacio en memoria para nuestro uso, y los datos primitivos se pueden crear de las siguientes formas:

* Sting *sNombre = “ ”;*
* Entero *iEdad ;*
* Const para declarar constantes *const float PI=3.1416*

*INICIO*

*mostrar “Dame tu nombre”;*

*aceptar (sNombre);*

*mostrar “Tu nombre es: “ + sNombre;*

*FIN*

Al representar **Diagramas de flujo** iniciamos con:

Inicio / Fin Entrada

Proceso Decisión

Flujo

Conectores

misma/dif Pag.

Salida

**ETC… ETC….**

**Tipos de datos:**

* Entero
* Doble (Decimal, Float, Flotante)
* Carácter ( “a” , ´b´)
* Cadena (“ ”)
* Lógico (booleano “true o false””1 o 0” )

*INICIO*

*Despertar;*

*Levantarse;*

*Bañarse;*

*Secarse el cabello;*

*Desayunar y Preparar Lonche;*

*Pintarse;*

*Cambiarse;*

*Preparar mi mochila;*

*Caminar al Autobús;*

*Tomarlo;*

*bajarme y caminar al salón;*

*FIN*

**EJERCICIOS**

**HPDF (Hacer Pseudocodigo y Diagrama de Flujo).**

* 1.- que sume dos números
* 2.- que le pida al usuario base y altura de un triángulo y regrese el área.
* 3.- que le pida al usuario su año de nacimiento y despliegue su edad.
* 4.- que pida al usuario un número y diga si es par o impar.
* 5.- pida al usuario su edad si su edad es menos que 5 mostrar “Niño” si se encuentra entre los 5 y los 18 “Adolecente” y si es entre 19 -35 Mostrar “ChavoRuco” y si es 36 mostrar “Inmortal” si es = o mayor de 37 “Viejito”.
* 6.- Pida el nombre al usuario y lo muestre 100 veces.
* 7.- Muestre la serie del 1 al 50.
* 8.- Muestre la serie de 50 a 1.
* 9.- Muestre la serie de 2 en 2 iniciando en 0 y terminando en 1000.
* 10.- de 3 en 3 hasta el 1000.
* 11.- de 5en 5 hasta el 1000.
* 12.- Pida un número (inicial y final) al usuario ascendente y descendente del número inicial al final de 1 en 1,3 en 3 y 5 en 5.
* 13.- Pedir al usuario un número y calcular si es primo.
* 14.- el usuario da un número y son los dígitos de la serie de Fibonacci que mostraremos.
* 15.- pida al usuario un numero le diga si es par o impar, le diga si es primo o no, y preguntarle si desea calcular otro número.
* 16.- Le pida al usuario una cadena y cuenta la cantidad de A, E, I, O, U.
* 17.- Pedir al usuario que introduzca una cadena, un número y mostrar la cadena, la cantidad de veces del número introducido.
* 18.-pedir al usuario un número y la potencia a la cual quiere elevar ese número.
* 19.- Sumar los números pares y primos del 100 al 200.
* 20.- Pida un número del 1 al 12 y muestre que mes corresponde.
* 21.- Que le pida al usuario un número y muestre la tabla de multiplicar del 1 al 20 de ese número.
* 22.- Pida al usuario 2 números y mostrar las tablas de multiplicar con esos límites. (el primero determina el límite de números a multiplicar.)
* 23.- Pedir un número y ese será el límite hasta donde será el Triángulo de pascal (1 22 333 4444 55555…)
* 24.- Pedir un número y calcular su factorial.
* 25.- Despliege Hola Mundo 1000 veces.
* 26.- Que pida la calificación en letra al alumno y de el equivalente A=10, B=9,C=8,D=7,E=6, F=5.

**(NOTA: Comprender bien tres cosas; descifrar entradas, procesos y salidas esperadas).**

**(NOTA2: Hacer cada ejercicio en Pseudocódigo y Diagrama de Flujo, Usando las 3 estructuras de ciclos; FOR, WHILE, DO WHILE).**

**1.- Ejemplo 1**

Dos entradas dos procesos una salida.

*entero iNum1 =0;*

*entero iNum1 =0;*

*entero iNum3 = 0;*

entero iRes = 0;

*entero iNum3 = 0;*

entero iRes = 0;

***INICIO***

*Mostrar “Ingresa un número”;*

Dame un numero

*Aceptar(iNum1);*

*Mostrar “Ingresa Otro número”;*

*Aceptar(iNum2);*

iNum1

*iRes=iNum1 + iNum2;*

*Mostrar “la suma es: “ + iRes;*

Dame otro número

***FIN***

iNum2

iRes = iNum1+iNum2

El resultado es iRes

**2.- Ejemplo 2**

Dos entradas, un proceso, una salida.

*entero iBase =0;*

*entero iAltura = 0;*

entero iRes = 0;

***INICIO***

*Mostrar “Ingresa la base”;*

*Aceptar(iNum1);*

*Mostrar “Ingresa la altura”;*

*Aceptar(iNum2);*

*iRes=iNum1 + iNum2;*

*Mostrar “El área: “ + iRes;*

***FIN***

**3.- Ejercicio 3**

Dos entradas, un proceso, una salida.

*entero iNum3 = 0;*

*entero iRes = 0;*

***INICIO***

*Mostrar “En que año naciste”;*

*Aceptar(iEdad);*

*iRes=iEdad + (-2018);;*

*Mostrar “Tu edad es”;*

***FIN***

**4.- Ejercicio 4**

Una entrada, un proceso, una salida.

*entero iNum1 = 0;*

***INICIO***

*Mostrar “Dame un número”;*

*Aceptar (iNum1);*

*Si iNum1 %2 ==0 entonces*

*Mostrar “Numero par”;*

*Caso Contrario*

*Mostrar “Número impar”;*

*Fin sí;*

***FIN***

**6.- Ejercicio 6.1 (for)**

Una entrada, un proceso, una salida.

*entero sName = ‘ ’;*

*entero iCont = 0;*

***INICIO***

*Mostrar “Dame tu nombre”;*

*Aceptar (sName);*

*For (entero contador = 1; contador < 100; contador ++1)*

*Mostrar “Tu nombre es fulanito”*

*Fin For*

***FIN***

**6.- Ejercicio 6.2 (while)**

Una entrada, un proceso, una salida.

*entero sName = ‘ ’;*

*entero iCont = 0;*

***INICIO***

*Mostrar “Dame tu nombre”;*

*Aceptar (sName);*

*While (iCont <= 100)*

*iCont = iCont + 1*

*Mostrar “Tu nombre es fulanito”*

*Fin While*

***FIN***

**6.- Ejercicio 6.2 (Do while)**

Una entrada, un proceso, una salida.

*entero sName = ‘ ’;*

*entero iCont = 0;*

***INICIO***

*Mostrar “Dame tu nombre”;*

*Aceptar (sName);*

*Do*

*iCont = iCont + 1*

*Mostrar “Tu nombre es fulanito”*

*While (iCont <= 100)*

*Fin While*

***FIN***

**7.- Ejercicio 7.1 (for)**

Una entrada, un proceso, una salida.

*entero iCont = 0;*

***INICIO***

*For (entero contador = 1; contador < 50; contador ++1)*

*Mostrar iCont;*

*Fin For*

***FIN***

**7.- Ejercicio 7.2 (while)**

Una entrada, un proceso, una salida.

*entero iCont = 0;*

***INICIO***

*While (iCont <= 50)*

*iCont = iCont + 1;*

*Mostrar iCont;*

*Fin While*

***FIN***

**7.- Ejercicio 7.3 (Do while)**

Una entrada, un proceso, una salida.

*entero iCont = 0;*

***INICIO***

*Do*

*iCont = iCont + 1;*

*Mostrar iCont;*

*While (iCont <= 49)*

*Fin While*

***FIN***

**8.- Ejercicio 8.1 (for)**

Una entrada, un proceso, una salida.

*entero iCont = 0;*

***INICIO***

*For (entero contador = 50; contador >=0; contador --1)*

*Mostrar iCont;*

*Fin For*

***FIN***

**8.- Ejercicio 8. (while)**

Una entrada, un proceso, una salida.

*entero iCont = 0;*

***INICIO***

*While (iCont => 0)*

*iCont = iCont - 1;*

*Mostrar iCont;*

*Fin While*

***FIN***

**8.- Ejercicio 8. (Do while)**

Una entrada, un proceso, una salida.

*entero iCont = 0;*

***INICIO***

*do*

*iCont = iCont - 1;*

*Mostrar iCont;*

*While (iCont => 0)*

*Fin While*

***FIN***

**Ejercicio 20**

***entero iMes = 0;***

***INICIO***

***Mostrar “Dame un numero”;***

***Aceptar (iMes);***

***(Con if´s seria)***

***if iMes == 1 Entonces***

***Mostrar “Enero”;***

***if iMes == 2 Entonces***

***Mostrar “Febrero”; …***

***(Con switch)***

***Caso 1: Mostrar “Enero”;***

***Break;***

***Caso 2: Mostrar “Febrero”;***

***Break;***

***Caso 3: Mostrar “Marzo”;***

***Break;***

***Caso 4: Mostrar “Abril”;***

***Break;***

***Caso 5: Mostrar “Mayo”;***

***Break;***

***Caso 6: Mostrar “Junio”;***

***Break;***

***Caso 7: Mostrar “Julio”;***

***Break;***

***Caso 8: Mostrar “Agosto”;***

***Break;***

***Caso 9: Mostrar “Septiembre”;***

***Break;***

***Caso 10: Mostrar “Octubre”;***

***Break;***

***Caso 11: Mostrar “Nov”;***

***Break;***

***Caso 12: Mostrar “Dic”;***

***Break;***

***Caso contrario “Mes no valido”;***

***fin switch;***

***FIN***

*Ejercicio 25:*

*INICIO*

*For (entero contador = 0; contador < 1000; contador +1)*

*Mostrar “Hola Mundo”*

*Fin For*

*FIN*

**TEMA 8**

**CICLOS**

***CICLO FOR***

For (contador = valor inicial ; Condición ; incremento/Decremento )

Acciones;

Fin For

en Diagrama de flujo se representa:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**<** cont=valor;cond; Inc/dec **>**

**En Pseudocodigo:**

While (condicion)

Accion…

Accion…

Fin While

**CICLO WHILE**

**CICLO DO WHILE**

Se usa el for cunado sabemos el número de veces a repetir el ciclo,

Cuando no sabemos cuántas veces vamos a repetir el ciclo.

**TEMA 9**

**JAVA**

**import** java.util.Scanner; // se debe importar la clase scanner para poder usar el objeto scanner.

Scanner entrada = **new** Scanner (System.***in***); //declarar variable

System.***out***.println("Dame tu nombre"); //mostrar

sName = entrada.nextLine(); //Aceptar

next es para una sola palabra.

nextLine es cuando se espera una cadena con varios espacios.

**char** iLetra = ' ' ;

//String cadena;

Scanner entrada = **new** Scanner(System.***in***);

System.***out***.println("Escribe una letra A-F ");

iLetra = entrada.next().charAt(0);

//cadena = entrada.next();

//iLetra =cadena.charAt(0);

System.out.println("La letra introducida fue: "iLetra);

**switch** (iLetra)

{

**case** 'A':

**case**'a': System.out.println("tienes 10 ");

**break**;

**case** 'B':

**case**'b': System.out.println("tienes 9 ");

**break**;

**case** 'C':

**case**'c': System.out.println("tienes 8 ");

**break**;

**case** 'D':

**case**'d': System.out.println("tienes 7 ");

**break**;

**case** 'E':

**case**'e': System.out.println("tienes 6 ");

**break**;

**case** 'F':

**case**'f': System.out.println("tienes 5 ");

**break**;

**default**: System.out.println("Calificacion no valida");

}

**TEMA 3**

**PROGRAMACION ORIENTADA A OBJETOS.**

Una clase es una plantilla (un cascaron) que te permite crear (objetos) estructuras de datos.

Es una estructura abstracta de lo que tú quieres.

A la creación de un objeto a partir de una clase se le llama instanciación de una clase o concretización de una clase ó creación de un nuevo objeto.

la palabra reservada para realizar esto es “new” en Java y C#.

Como creamos una clase:

|  |
| --- |
| ***Libro*** |
| ***+*** *ISBN: String* |
| ***-*** *Titulo: String* |
| ***#*** *Prestar(): Boolean* |
| *Eliminar(): void* |
|  |

***class Libro*** *{*

*String ISBN;*

*String Titulo;*

*Boolean Prestar(){*

*}*

*void Eliminar(){*

*//void no regresa ningún valor*

*}*

*}*

*.*

Entre las clases podemos crear jerarquías a través de diferentes mecanismos.

En semántica son denominadas axioma (estructura de clase).

Modificadores de accesibilidad:

Son:

* **+** “públic” los miembros de esta clase son accesibles al mundo exterior.
* **-**  “prívate” los miembros de esta clase solo son accesibles al interior de la clase.
* **#**  “protected” los miembros de esta clase son accesibles por la clase y sus subclases.
* “static” los miembros le pertenecen a la clase. (subrayado).

El nombre de tu archivo debe ser el mismo que el de tu clase si no compila.

Al compilar, el compilador buscara dentro del mismo package (directorio) todos los .Java (.clas) a los que nosotros nos referimos.

appTinderLibreria

package com.apptinderlibreria.main

import com.apptinderlibreria.libro;

clas AppTinderLibreria

{

Libro objLibro1; //no esta creado solo es un espacio en la memoria aun.

obj Libro1 = **new** Libro();

}

Métodos Constructores y Clases.

Los constructores (son métodos) sirven para inicializar los valores del objeto.

Los métodos constructores son públicos.

Los métodos constructores son void.

Tienen que tener el mismo nombre de la clase de la que fueron formadas.

Todas las clases creadas en java generar un método constructor vacio.

Cantidad y tipo de parámetros colocado es como se diferencian un método constructor de otro.

Se pueden crear dos métodos con el mismo nombre pero deben variar en cantidad o tipo de parámetros para no caer en sobrecarga de métodos.

Métodos destructores.

Creados por default.

Dependiendo el lenguaje es nombre de la clase ( **\_** , **~**, **[]** )

Si lo creamos nosotros es pasando a null todas las propiedades.

(al poner en null nuestrob garbash collector lo elimina) “objLibro1.close();”

INTELLISENSE

Autocompletado.

\*\*\*leer sobre getters y setters\*\*\*\*

Modificadores de accesibilidad

encapsuladores.

i

**API**

Application Programming Interface.

Conjunto de clases contenidas en un package que dan funcionalidad al Sistema.

**Herencia Multiple.**

java no permite herencia multiple a menos que sea con interfaces.

**Interfaz** especifica lo que debe ser un objeto pero no le dice como

**clase abstracta** es una clase que no púede ser instanciada.

**TEMA 4**

**ARRAYS**

* HDFPque cree un vectpore de 100 elementos y los rellen con 1.
* HDFPcree un vector de 100 elementos y lo rellene con la serie 1,2,3,4,5..++
* HDFP cree un vector de 100 elementos y lo rellene con la serie 2,4,6,8,10..2+
* HDFP 100 elementos enteros y muestre la derie de 100,99,98,97.. - -
* HDFP 100 elementos random de el 1 al 100.
* HDFP de 100 numeros aleatorios que diga cuantos son pares e imapres.
* HDFP de 100 numeros aleatorios y diga cuales son numeros primos.
* HDFP de 100 numeros aleatorios y diga cuales y cuantos son numeros primos.
* HDFP vector de 100 elementos aleatorios y relize la suma todos los numeros.

**Los metodos deben de tener la maxima cohesion y el minimo acoplamiento.**

**\*\*\*\*POJO\*\*\*\***

**Plain Old Java Object**

**PILA = FILO, PUT & GET,**

**COLA= FIFO, VECTOR LIST, PUT&GET**