



UNIVERSIDAD SANTO TOMÁS

PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA

SECCIONAL TUNJA

VIGILADA MINEDUCACIÓN - SNIES 1732



Acreditación Institucional
Internacional

OTORGADA POR EL IAC CINDE ACUERDO 55 DEL 8 DE MAYO-VIGENCIA 5 AÑOS



Vigencia por seis años





UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
SECCIONAL TUNJA

VIGILADA MINEDUCACIÓN - SNIES 1732

Faculty: Systems engineer
Course: Introduction of Programming
Topic: Fundamentals of software programming

Socializer: Luis Fernando Castellanos Guarín
Email: Luis.castellanosg@usantoto.edu.co
Phone: 3214582098



Topics

- **Course Introduction**
 - What is the software? / Que es el software?
 - Programming languages / Lenguajes de programación
- **History of computing**
- **Concepts and elements of a Flowcharts**

¡Siempre
hacia lo alto!





Course Introduction

¡Siempre
hacia lo alto!



What is the software? / ¿Qué es el software?

The software is a set of instructions that a computer runs. A program can have a few lines of code in a single file and run on a computer or millions of instructions that are in miles of files and are executed by hundreds of computer equipment.

El software es un conjunto de instrucciones que son ejecutadas por un computador. Un programa puede tener unas pocas líneas de código en un único archivo y ejecutados en un computador o millones de instrucciones que se encuentran en miles de archivos y son ejecutados por cientos de equipos de cómputo.

¡Siempre
hacia lo alto!



Programming languages



01101000 01101111 01101100 01100001
00100000 01100010 01110101 01100101
01101110 01101111 01110011 00100000
01100100 11000011 10101101 01100001
01110011



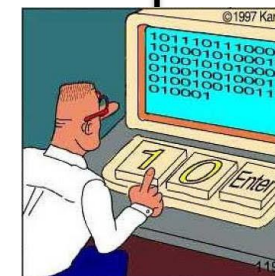
“hola buenos días”

las computadora solo pueden entender en binario (0 y 1)

¡Siempre
hacia lo alto!



Programming languages: Abstraction level



¡Siempre
hacia lo alto!



Programming languages: Abstraction level

Low level: example
assembler program

Hola mundo / hello world

```
.global _start

.text
_start:
    # write(1, message, 13)
    mov    $1, %rax      # system call 1 is write
    mov    $1, %rdi      # file handle 1 is stdout
    mov    $message, %rsi # address of string to output
    mov    $13, %rdx     # number of bytes
    syscall              # invoke operating system to do the write

    # exit(0)
    mov    $60, %rax     # system call 60 is exit
    xor    %rdi, %rdi    # we want return code 0
    syscall              # invoke operating system to exit
message:
    .ascii "Hello, world\n"
```

Potencia de un número

```
name "potencia" ;8 chars DOS
org 100h ;counter to 100h
mov cx, num2
mov ax, num1
inicio:
    mov bx,num1
    mul bx ;ax = ax * bx
    loop inicio ;c-
mov num3,ax ;copiamos el resultado
ret ;
Variables "db" para byte y "dw" para word
num1 dw 0Ah
num2 dw 03h
num3 dw 0h
```

¡Siempre
hacia lo alto!



Programming languages: Abstraction level

Medium level:
example C program

Hola mundo / hello world

```
#include <stdio.h>

int main() {
    printf("Hola mundo");
    return 0;
}
```

Potencia de un número

```
#include <stdio.h>
#include <stdlib.h>

int main() {
    int num, i;
    i=1;
    printf("Introduzca un número entero: ", 163);
    scanf("%d", &num);
    while(i<=20){
        printf("El resto de %d entre %d es: %d\n", i, num, i%num);
        i=i+1;
    }

    return 0;
}
```

¡Siempre
hacia lo alto!



Programming languages: Abstraction level

High level: example
JAVA+PYTHON+PHP

Hola mundo / hello world

Java

```
public class MyClass {  
  
    public static void main(String args[]) {  
        System.out.println("Hola mundo");  
    }  
}
```

Python

```
println("Hola mundo")
```

Php

```
<html>  
<head>  
    <title>Prueba de PHP</title>  
</head>  
<body>  
    <?php echo '<p>Hola Mundo</p>'; ?>  
</body>  
</html>
```

Ventajas:

- Mejor comprensión del lenguaje
- Independiente del equipo

¡Siempre
hacia lo alto!



Compiled languages vs. interpreted / Lenguajes compilados vs. interpretados

When does the source code become binary?

Some languages require that all your code be compiled before any of the codes can be executed; other languages interpret each instruction at the time the code is executed.

When you are writing a program in a compiled language, you must compile the program after each change in the source code before running the program again.



¿Cuándo se convierte el código fuente en binario?

Algunos idiomas requieren que se compile todo su código antes de que se pueda ejecutar cualquiera de los códigos; otros idiomas interpretan cada instrucción en el momento en que se ejecuta el código.

Cuando está escribiendo un programa en un lenguaje compilado, debe compilar el programa después de cada cambio en el código fuente antes de volver a ejecutar el programa.

¡Siempre
hacia lo alto!



P1T3_LINGÜAJES DE PROGRAMACIÓN MÁS USADOS

En grupos de máximo dos(2) personas, hacer una investigación sobre uno de los lenguajes de programación mas usados en el 2019 (JAVASCRIPT, SWIFT, JAVA, C/C++, C#, PYTHON, PHP, R, Objective-C, Matlab, Kotlin, Go), donde explique:

- Principales características
- Ventajas del lenguaje
- Desventajas del lenguaje
- Que empresas internacionales lo usan

Hacer una presentación usando la plantilla institucional (2020), teniendo presente las siguientes recomendaciones:

- Máximo 5 diapositivas
- Máximo 5 renglones de texto por diapositiva.
- Toda imagen y texto copiado de internet debe tener su webgrafía.



Programación = matemáticas

A decorative geometric pattern in the bottom right corner, consisting of overlapping squares and rectangles in blue, red, and yellow colors.

¡Siempre
hacia lo alto!



Arithmetic operations / Operaciones aritméticas

En la programación de software existen muchas operaciones aritméticas a continuación se mencionan las más comunes en los lenguajes de operación.

Generalmente nos permiten tratar con números y, como su nombre lo dice, **realizar operaciones aritméticas**:


- Para sumar, se usa el operador de suma (+) : $4 + 5$, También es posible sumar cadenas de texto de la siguiente forma: "Hola " + "Mundo"
- Para restar, usamos el operador de resta (-) : $4 - 5$
- Para multiplicación se usa como operador el asterisco (*)
- Para división se usa como operador la diagonal (/).

Además, tenemos un último, llamado **módulo**, que se representa con el símbolo del porcentaje (%). Este nos regresa el **residuo de una división** entre dos números



Operations Hierarchy / Jerarquía de operaciones

Regla general de la Jerarquía de Operaciones:

1ª	Raíces Cuadradas o Potencias		
2ª	Paréntesis o Corchetes		
3ª	Multiplicación (producto) o División		
4ª	Sumas o Restas		
5ª	Dirección: de Izquierda a Derecha		



P1T4: Exercise of Operations Hierarchy /

Jerarquía de operaciones

Resuelva las siguientes operaciones, explicando el paso a paso (como un algoritmo):

- $2 + 5 * 4 = 22$
- $9 - 7 + 5 + 2 - 6 + 8 - 4 = 7$
- $3 * 2 - 5 + 4 * 3 - 8 + 5 * 2 = 15$
- $(15 - 4) + 3 - (12 - 5 * 2) + (5 + 16 / 4) - 5 + (10 - 2^3) = 801$



¿Cómo funciona un software?

¡Siempre
hacia lo alto!



How does the software work? / Como funciona un software?

Las computadoras son las maquinas mas estúpidas que existen “Solo hacen lo que el programador le dijo que hiciera”

Software:

“Soporte lógico de un sistema informático, que comprende el conjunto de los componentes lógicos necesarios que hacen posible la realización de tareas específicas, en contraposición a los componentes físicos que son llamados hardware”.. Wikipedia

“Un conjunto de instrucciones que transforma el código escrito en un lenguaje de programación a algo que la computadora puede ejecutar”... Profesor

Posdata:

“Si mis instrucciones tienen errores (gramaticales, semánticos o tipográficos), el lenguaje de programación no podrá traducir correctamente a binario”



What is an algorithm? / ¿Qué es un algoritmo?

An algorithm is a sequence of logical steps necessary to carry out a specific task, such as solving a problem. The algorithms are independent of both the programming language in which they are expressed and the computer that executes them. In each problem the algorithm can be expressed in a different programming language and run on a different computer; however, the algorithm will always be the same

Un algoritmo es una secuencia de pasos lógicos necesarios para llevar a cabo una tarea específica, como la solución de un problema. Los algoritmos son independientes tanto del lenguaje de programación en que se expresan como de la computadora que los ejecuta. En cada problema el algoritmo se puede expresar en un lenguaje diferente de programación y ejecutarse en una computadora distinta; sin embargo, el algoritmo será siempre el mismo



MK1, el robot chef que cocina lo que quieras

*Por ejemplo, en una analogía con la vida diaria, **una receta de cocina** se puede expresar en **español, inglés o francés**, etc. Los pasos para la elaboración del plato se realizarán sin importar el cocinero ni el lenguaje.*

**¡Siempre
hacia lo alto!**

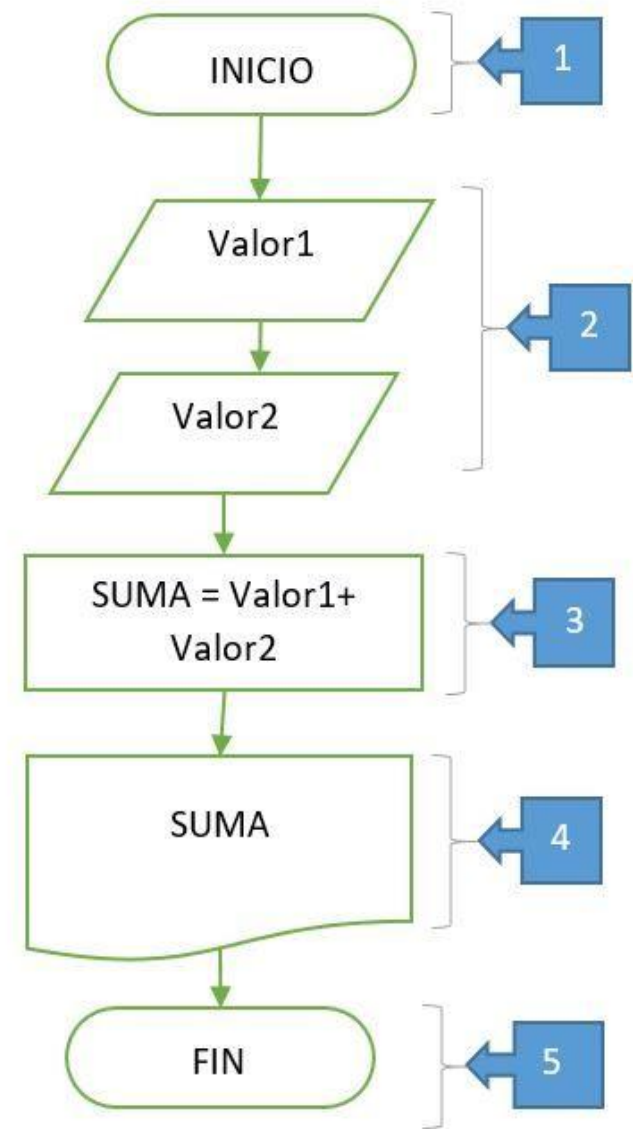


Flowchart / Diagrama de flujo

Un diagrama de flujo u organigrama es una representación diagramática que ilustra la secuencia de las operaciones que se realizarán para conseguir la solución de un problema.

Los diagramas de flujo se dibujan generalmente antes de comenzar a programar el código frente a la computadora.

Estos diagramas de flujo desempeñan un papel vital en la programación de un problema y facilitan la comprensión de problemas complicados y sobre todo muy largos. Una vez que se dibuja el diagrama de flujo, llega a ser fácil escribir el programa en cualquier idioma de alto nivel. (miunicobloginformatico, 2010).

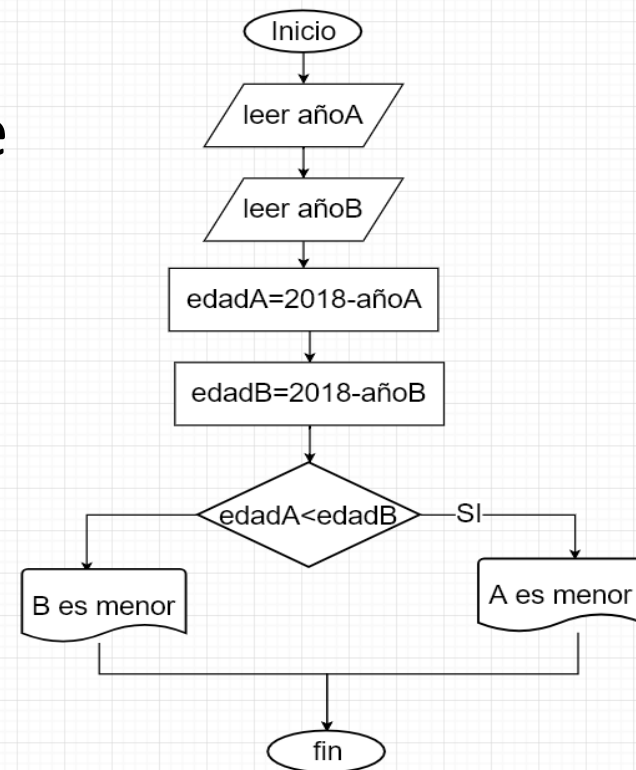




Flowchart / Diagrama de flujo

Ejemplo de algoritmo representado mediante un diagrama de flujo:

Que reciba el año de nacimiento de la persona A y el año de nacimiento de la persona B e imprima un mensaje indicando cuál de las dos personas es menor



El uso de los diagramas de flujo para documentar procesos de negocios se inició entre las décadas de 1920 y 1930

¡Siempre
hacia lo alto!



Elements of Flowchart / Elementos de diagrama de flujo

SIMBOLOS FUNDAMENTALES



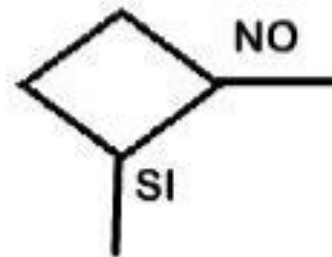
INICIO Y FIN DEL PROCESO



REALIZAR UN PROCESO (OPERACIÓN MATEMÁTICA POR EJEMPLO)



ENTRADA DE DATOS Y/O SALIDA DE DATOS



TOMAR UNA DECISIÓN (UNA PREGUNTA).
LA RESPUESTA A LA PREGUNTA
PUEDE SER SI O NO

¡Siempre
hacia lo alto!



Basic variables in programming / variables básicas en programación

Tipo de dato		Ejemplo
short	Números cortos	20
Int	Números enteros	4556
long	Números enteros largos	4566566
float	Números flotantes	7.545601
double	Número dos veces más grande que float	7.5456015333333333
byte	Manejo de 0 y 1	001110
char	Carácter	'a'
boolean	Booleanos verdad o falso	False
String	Cadenas de caracteres	"Juan David Peña"

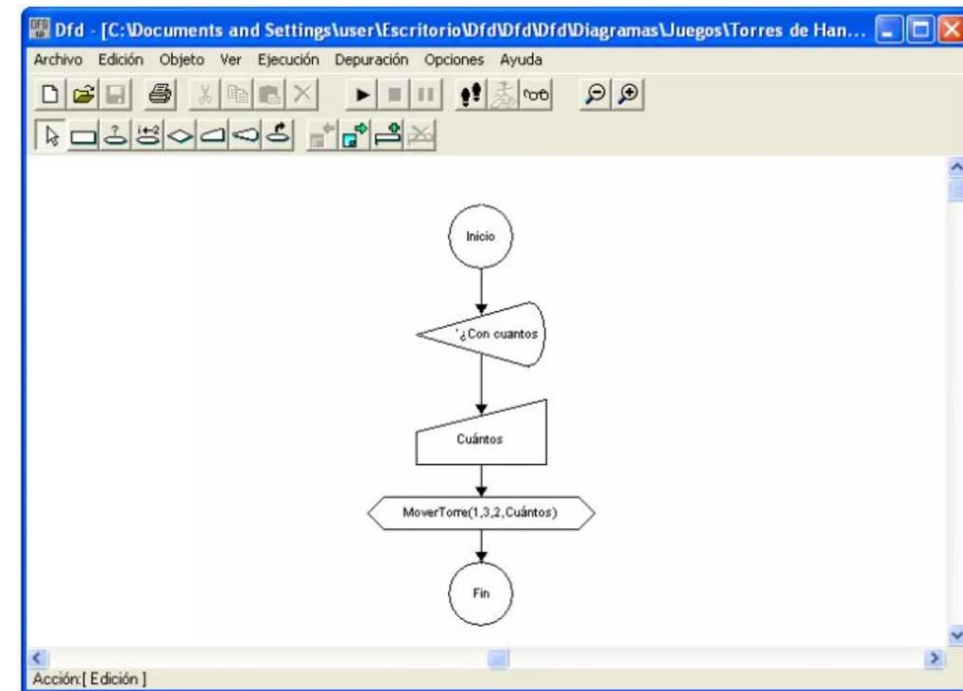


DFD – Description / descripción del DFD

Free software with available code (GPL)
Translated into English and Portuguese
whose functionality is to be a flowchart
editor with which you can graphically form a
large number of algorithms, execute them
and debug them in case of errors

Software libre con código disponible (GPL)
Traducido al inglés y portugués cuya
funcionalidad es ser un editor de diagramas
de flujo con el cual puedes dar forma gráfica
a un gran número de algoritmos, ejecutarlos
y depurarlos en caso de hallar errores

- https://github.com/luisFernandoCastellanosG/Introduction_to_programming/Software



Grupo Smart, de la Universidad del Magdalena
(Santa Marta, Colombia)

¡Siempre
hacia lo alto!



Exercises in DFD

**LET'S
GO!**

¡Siempre
hacia lo alto!



P1T5: adding numbers

Design an algorithm using DFD where:

Enter the value of three numbers and sums by keyboard.

Diseñe un algoritmo usando DFD donde:

Ingresa por teclado el valor de tres números y súmelos.



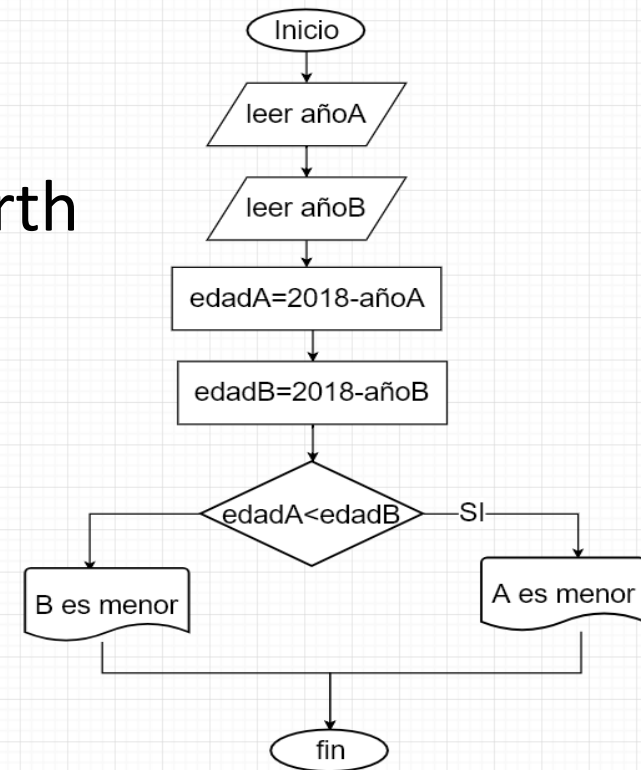
P1T6: compare birth of two person

Design an algorithm using DFD where:

Enter the year of birth of person A and the year of birth of person B on the keyboard and print a message indicating which of the both persons is minor

Diseñe un algoritmo usando DFD donde:

Ingrese por teclado el año de nacimiento de la persona A y el año de nacimiento de la persona B e imprima un mensaje indicando cuál de las dos personas es menor





P1T7: Pay point

Design an algorithm using DFD where:

The supermarket "paradise" requires building a **payment point software**. The software receives 4 data that corresponds to the value of each of the products registered by a customer, the software must add the values of the four products and add "IVA" of 19%. In the invoice the customer can observe the value of the products, the value of "IVA" and the total to be paid.

Diseñe un algoritmo usando DFD donde:

El supermercado "paraíso" requiere construir una aplicación de punto de pago. La aplicación recibe 4 datos que corresponde al valor de cada uno de los productos registrados por un cliente, la aplicación deberá sumar los valores de los cuatro productos y agregar el IVA de 19%. En la factura el cliente podrá observar el valor de los productos, el valor del IVA y el total a pagar.



P1T8: "Blood pressure"

Design an algorithm using DFD, where:

Build an algorithm that, upon receiving the systolic and diastolic pressure data, indicates whether the patient has hypertension or normal blood pressure.

Diseñe un algoritmo usando DFD, donde:

Construya un algoritmo que, al recibir los datos de presión sistólica y diastólica, indique si el paciente tiene hipertensión o presión arterial normal.



Clasificación de PA	Sistólica mmHg	Diastólica mmHg
Normal	<120	<80
Prehipertensión	120-139	80-89
Hipertensión Estadio 1	140-159	90-99

¡Siempre hacia lo alto!



P1T9: Weekday

Design an algorithm using DFD, where:

When entering a number the program must print the day of the week in letters.

Diseñe un algoritmo usando DFD, donde:

Al ingresar un número el programa debe imprimir el día de la semana en letras.



P1T10: Semester grades

Design an algorithm using DFD, where: Perform an algorithm that when you enter a student's grades in the semester, the grades have the following weights:

- 1st exam 20%
- 2nd exam 25%
- 3rd exam 25%
- 4th exam 30%

With the notes and the weighted calculate the final semester grade

Diseñe un algoritmo usando DFD, donde: Realice un algoritmo que cuando ingresa las calificaciones de un estudiante en el semestre, las calificaciones tienen los siguientes pesos:

- 1er examen 20%
- 2º examen 25%
- 3er examen 25%
- 4to examen 30%

Con las notas y la ponderación calcule la calificación final del semestre



P1T11: hypotenuse

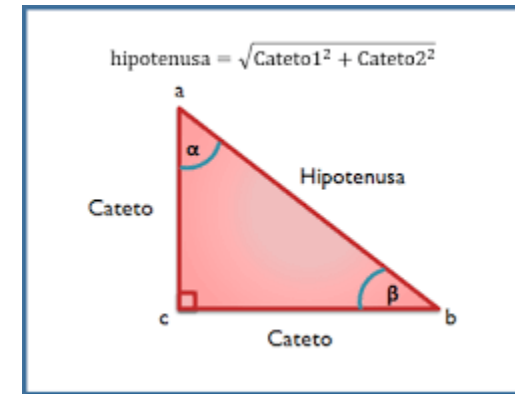
Design an algorithm using DFD, where:

Make a program to calculate the hypotenuse of a triangle.

$$\sqrt{x} = \text{Sqrt}(x)$$

Diseñe un algoritmo usando DFD, donde:

Realice un programa para calcular la hipotenusa de un triángulo.



Siempre
hacia lo alto!



P1T12: Age of people

Design an algorithm using DFD, where:

with the age of a person determine if he is a child (<6) adolescent (<18) adult (<45) older adult (<90) matusalen ≥ 90

Diseñe un algoritmo usando DFD, donde:

con la edad de una persona determinar si es niño (<6) adolescente (<18) adulto (<45) adulto mayor (<90) matusalen ≥ 90



P1T13: volumen of cylinder

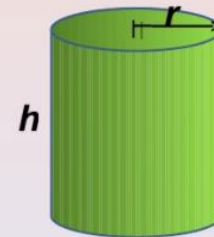
Design an algorithm using DFD, where:

create an algorithm that calculates the volume of a cylinder knowing the radius and height.

Diseñe un algoritmo usando DFD, donde:

crear un algoritmo que calcule el volumen de un cilindro conociendo el radio y la altura.

▪Volumen del cilindro



Volumen: V

$$V = A_B (h)$$

$$V = \pi r^2 h$$

r : radio del cilindro

h : altura del cilindro

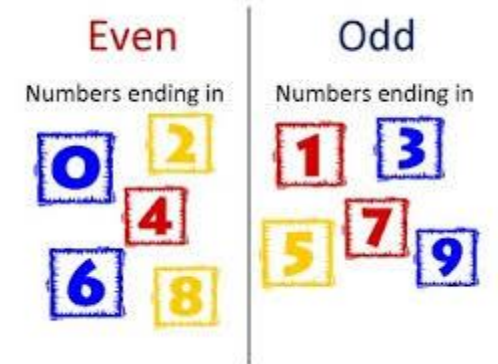
¡Siempre
hacia lo alto!



P1T14: number even or odd

Design an algorithm using DFD, where:

Entering a number indicate if it is even or odd



Diseñe un algoritmo usando DFD, donde:

Ingresando un número indique si es par o impar

¡Siempre
hacia lo alto!



P1T15: largest of three numbers

Design an algorithm using DFD, where:

Determine which is the largest of three numbers

Diseñe un algoritmo usando DFD, donde:

Determinar cual es el mayor de tres números

¡Siempre
hacia lo alto!



P1T16: Calendar

Design an algorithm using DFD, where:

with the number of the month determine how many days the month has

note: the month number must be between 1 and 12



Diseñe un algoritmo usando DFD, donde:

con el número del mes determine cuántos días tiene el mes

nota: el número del mes debe estar entre 1 y 12





P1T17: salary of an employee

Design an algorithm using DFD, where:

Calculate the salary of an employee taking into account the salary hired and the days worked in the month.

note:

Days cannot be less than 0 or greater than 30

Diseñe un algoritmo usando DFD, donde:

Calcular el salario de un empleado teniendo presente el sueldo contratado y los días trabajados en el mes

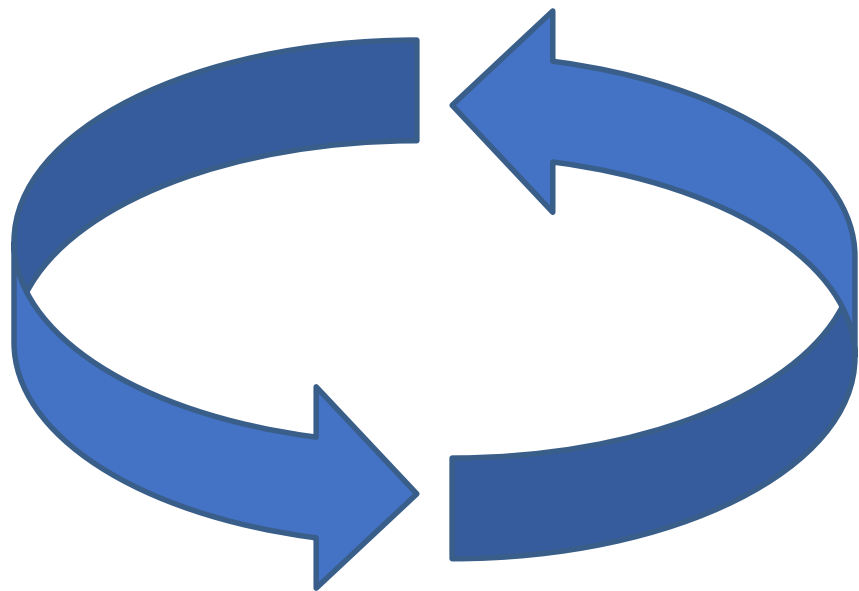
nota:

Los días no pueden ser menores de 0 ni mayores de 30





Cycles “for” and “while”



Design an algorithm using DFD, where:

display N numbers using the "**cycle for**"

Diseñe un algoritmo usando DFD, donde:

visualice N números usando el ciclo para



Exercise in DFD - P1T19: total sum for N numbers

Design an algorithm using DFD, where:

display the total and average sum for N numbers using the "cycle for"

Diseñe un algoritmo usando DFD, donde:

visualice la suma total y promedio para N números usando el “ciclo para”



UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
T U N J A
VIGILADA MINEDUCACIÓN - SNIES 1722



Formando personas que transforman



Exercise in DFD - P1T20: grade point average

Design an algorithm using DFD, where:

calculate the average grade of the course where the final grade of each student is the
grade1 = 20%, grade2 = 30% and grade3 = 50%

Diseñe un algoritmo usando DFD, donde:

calcule la nota promedio de curso donde la nota final de cada estudiante es el nota1
=20%, nota2=30% y nota3=50%



UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
T U N J A
VICELADIA MINEDUCACIÓN - SNIES 1722



Formando personas que transforman



Exercise in DFD - P1T21: Product invoice N

Design an algorithm using DFD, where:

calculate gross value and total value = gross value + VAT (19%) for N products

Diseñe un algoritmo usando DFD, donde:

calcular el valor_bruto y valor_total = valor_bruto+IVA(19%) para N productos



UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
T U N J A
VICELADIA MINEDUCACIÓN - SNIES 1722



Formando personas que transforman



Exercise in DFD - P1T22: salary of many employee

Design an algorithm using DFD, where:

Calculate the **average_salary**, **total_salary** and **highest_salary** of **N** employees, where
 $\text{salary} = (\text{salary} / 30) * \text{working_days}$

Diseñe un algoritmo usando DFD, donde:

Calcular el salario promedio, salario total y salario más alto de N empleados, donde
 $\text{salario} = (\text{suelto} / 30) * \text{dias_trabajados}$



UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
T U N J A
VIGILADA MINEDUCACIÓN - SNIES 1722



Formando personas que transforman



Exercise in DFD - P1T23: odd numbers from 11 to 99

Design an algorithm using DFD, where:

make a program that prints odd numbers from 11 to 99

Diseñe un algoritmo usando DFD, donde:

realice un programa que imprima los números impares desde 11 hasta 99



UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
T U N J A
VIGILADA MINEDUCACIÓN - SNIES 1722



Formando personas que transforman



Exercise in DFD - P1T24: minors and greater people

Design an algorithm using DFD, where:

make a program that prints how many are minors and how many are of legal age for N people

Diseñe un algoritmo usando DFD, donde:

realice un programa que imprima cuantos son menores de edad y cuantos son mayores de edad para N personas



UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
T U N J A
VIGILADA MINEDUCACIÓN - SNIES 1722



Formando personas que transforman



Design an algorithm using DFD, where:

with the ages of N person determine how many are children (<6)
adolescents (<18) adults (<45) older adults (<90) matusalen ≥ 90

Diseñe un algoritmo usando DFD, donde:

con la edades de N personas determinar cuantos son niños (<6)
adolescentes (<18) adultos (<45) adultos mayores (<90) matusalen ≥ 90



Exercise in DFD - P1T26: route of a bus

Design an algorithm using DFD, where:

Make a program that simulates the route of a bus, at each stop many passengers get on and off, determine how many passengers were transported and how much money was obtained if each ticket is worth \$ 150

Diseñe un algoritmo usando DFD, donde:

Haga un programa que simule la ruta de un autobús, en cada parada entran y salen muchos pasajeros, determine cuántos pasajeros fueron transportados y cuánto dinero se obtuvo si cada boleto vale \$ 150



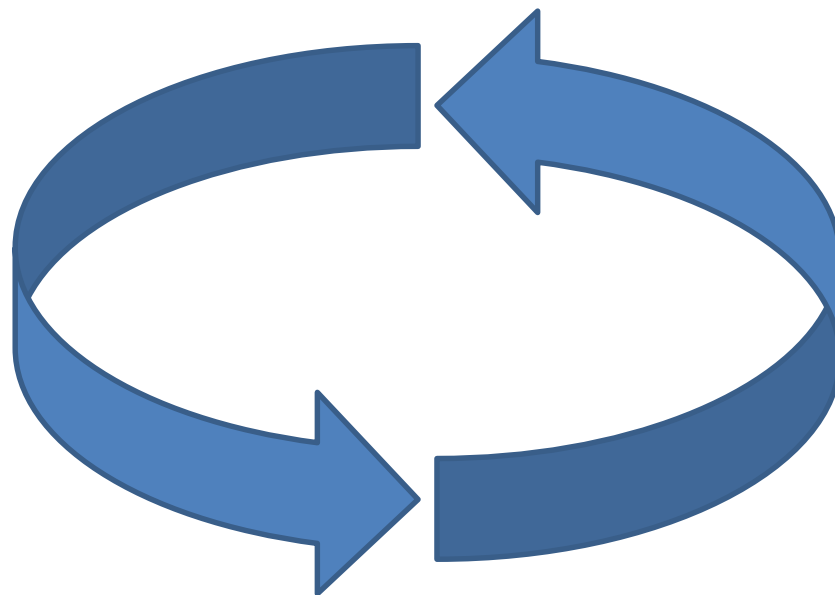
UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
T U N J A
VICELADIA MINEDUCACIÓN - SNIES 1722



Formando personas que transforman



Cycles “while”



UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
T U N J A
VIGILADA MINEDUCACIÓN - SNIES 1722



Formando personas que transforman



Exercise in DFD - P1T27: lower than 100

Design an algorithm using DFD, where:

Read from the keyboard a series of numbers until you get one lower than 100, in the end determine the sum and average of the numbers entered

Diseñe un algoritmo usando DFD, donde:

Leer desde el teclado una serie de números hasta obtener uno inferior a 100, al final determinar la suma y promedio de los números ingresados.



UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
T U N J A
VIGILADA MINEDUCACIÓN - SNIES 1722



Formando personas que transforman



Exercise in DFD - P1T28: market of the month

Design an algorithm using DFD, where:

A housewife needs to make the market of the month and only has \$ 500,000 pesos. Perform an algorithm that captures the price of each product and adds it until the account reaches the maximum limit (500,000), in the end determine how many products you buy and when it was the total value you paid

Diseñe un algoritmo usando DFD, donde:

Un ama de casa requiere hacer el mercado del mes y solo tiene \$500.000 pesos. Realizar un algoritmo que capture el precio de cada producto y lo sume hasta que la cuenta llegue al tope máximo (500.000), al final determine cuantos productos compro y cuanto fue el valor total que pago.



UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
T U N J A
VIGILADA MINEDUCACIÓN - SNIES 1722



Formando personas que transforman



Design an algorithm using DFD, where:

A software that records the times of each lap of training and determines the best lap, the average and the number of laps per training (the laps will be until the athlete gets tired / exhausted)

Diseñe un algoritmo usando DFD, donde:

Un software que me registre los tiempos de cada vuelta de entrenamiento y me determine la mejor vuelta, el promedio y el número de vueltas por entrenamiento (las vueltas serán hasta que se canse/agote el deportista)



Exercise in DFD - P1T30: password

Design an algorithm using DFD, where:

Ask for a password (3 attempts maximum) or until the password is 352, 259 or 569.

Diseñe un algoritmo usando DFD, donde:

Pedir una contraseña (3 intentos máximo) o hasta que la clave sea 352, 259 ó 569.



UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
T U N J A
VICELADIA MINEDUCACIÓN - SNIES 1722



Formando personas que transforman



Exercise in DFD - P1T31: calculator

Design an algorithm using DFD, where:

Simulation of a calculator is performed, depending on the following options:

- (1) Add two numbers.
- (2) Subtract two numbers.
- (3) Multiply two numbers.
- (4) Divide two numbers
- (0) to exit

Diseñe un algoritmo usando DFD, donde:

Se realice la Simulación de una calculadora, dependiendo de las siguientes opciones :

- (1) Sumar dos números.
- (2) Restar dos números.
- (3) Multiplicar dos números.
- (4) Dividir dos números
- (0) para salir

Formulario:

<https://forms.gle/9jcAdc3katSzo8w7A>



UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
T U N J A
VICELADIA MINEDUCACIÓN - SNIES 1722



Formando personas que transforman



Procedures or functions



UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
T U N J A
VIGILADA MINEDUCACIÓN - SMES 1722



Formando personas que transforman



Exercise in DFD - P2TX: calculator In JAVA

Make a program in JAVA, where:

Simulation of a calculator is performed, depending on the following options:

- (1) Add two numbers.
- (2) Subtract two numbers.
- (3) Multiply two numbers.
- (4) Divide two numbers
- (0) to exit

Cree un programa en JAVA, donde:

Se realice la Simulación de una calculadora, dependiendo de las siguientes opciones :

- (1) Sumar dos números.
- (2) Restar dos números.
- (3) Multiplicar dos números.
- (4) Dividir dos números
- (0) para salir



Exercise in DFD - P2TX: calculator In JAVA

Make a program in DFD/JAVA, using functions (the menu must dinamic):

Write a program that simulates the home delivery software menu, example: rappi, domicilios, glovo, lfood, Uberfood. our software is called USTArappi and it works only in Tunja:

Depending on the GPS position the price and the delivery time address changes:

1. Miscos (10 minutes and \$5000)
2. Center (5 minutes and \$2000)
3. Hongos (15 min and \$7500)
4. Las quintas (10 min and \$4000)
5. San Francisco (20 min and \$10.000)

The services available are:

1. Purchases in supermarkets (\$15.000)
2. Payment of invoices in banks (\$10.000)
3. Carrying objects (\$5.000)

The software must indicate the total price and the time of the service



Exercise in DFD - P2TX: calculator In JAVA

Make a program in JAVA, using functions (the menu must dinamic):

Write a program that simulates book loans in a library:

1. loan
2. Return

If it is a loan, the software must generate another menu where:

1. Books (maximum 4 days)
2. Journals (maximum 3 days)
3. Videos (maximum 2 days)
4. Computers (maximum 1 day)

If it is a Return, the software must generate another menu where:

1. Books
2. Journals
3. Videos
4. Computers

and must indicate the date of the loan and with that data define if the user must pay a fine.



Contenido	completado
Fundamentos de la programación de software	YES
Introducción al uso de herramientas CASE para programación. Conceptos del uso de herramientas drag&drop en programación	YES
Estructura condicional en un diagrama y su símil en código Java y PYTHON	YES
Estructura condicional simple y compleja de uso de IF	YES
Ciclo FOR, diagramas y su representación en código	YES
Ciclo FOR con uso de sentencias IF o uso anidado.	YES
Ciclo WHILE y su uso con otras estructuras condicionales.	YES
Ejercicios y talleres con ciclos While, FOR, estructuras condicionales y estructuras de opciones	YES
Programación de soluciones a problemas matemáticos usando ciclos.	YES
Funciones y procedimientos en la programación de software	YES
Manejo de información usando arreglos unidimensionales	
Operaciones matemáticas simples con vectores	
Operaciones avanzadas con vectores.	
Arreglos multidimensionales simples	
Operaciones avanzadas con arreglos multidimensionales	
Persistencia de datos usando archivos de texto.	
Implementación de un software con operaciones CRUD	



Arreglos Unidimensionales

Un arreglo(Array) es un medio de guardar un conjunto de objetos de la misma clase.

Se accede a cada elemento individual del array mediante un número entero denominado índice. 0 es el índice del primer elemento y $n-1$ es el índice del último elemento, siendo n , la dimensión del array. Los arrays son objetos en Java y como tales vamos a ver los pasos que hemos de seguir para usarlos convenientemente



UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
T U N J A
VICELADIA MINEDUCACIÓN - SNIES 1722



Formando personas que transforman

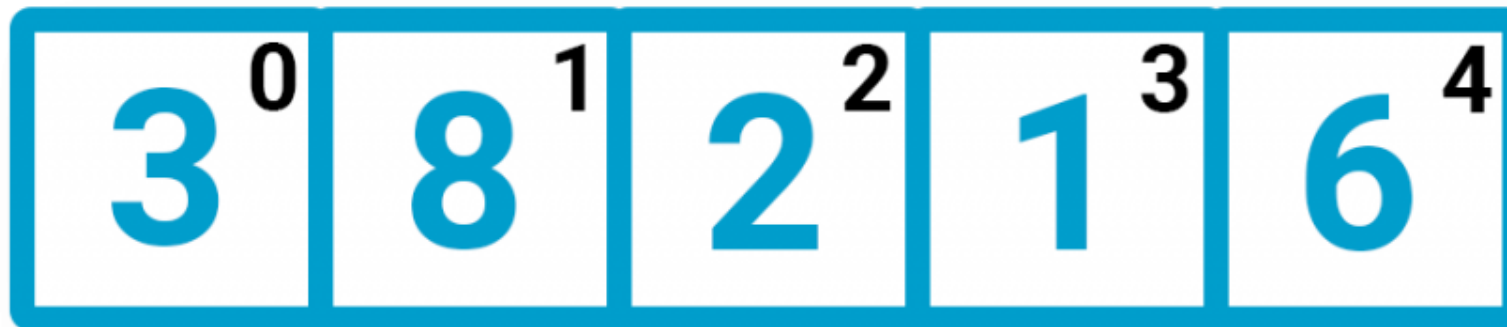


Array (Arreglos)

Ejemplo: Arreglo de números enteros:



Dentro del array y las posiciones para obtener esos números. Como vemos va desde cero (0) hasta cuatro (4)



Array (Arreglos)



La sintaxis para declarar e inicializar un array será:

```
Tipo_de_variable[ ] Nombre_del_array = new Tipo_de_variable[dimensión];
```

El tipo de variable puede ser cualquiera de los admitidos por Java y que ya hemos explicado. Ejemplos de declaración e inicialización con valores por defecto de arrays usando todos los tipos de variables Java, serían:

```
Tipo_de_variable [ ] Nombre_del_array;  
Nombre_del_array = new Tipo_de_variable[dimensión];
```

Ejemplo:

```
- byte[ ] edad = new byte[4];  
- short[ ] edad = new short[4];  
- int[ ] edad = new int[4];  
- long[ ] edad = new long[4];  
- float[ ] estatura = new float[3];  
- double[ ] estatura = new double[3];  
- boolean[ ] estado = new boolean[5];  
- char[ ] sexo = new char[2];  
- String[ ] nombre = new String[2];
```

Array (Arreglos)



Para números enteros

```
int[ ] edad = {45, 23, 11, 9}; //Array de 4 elementos
```

Para números reales

```
double[ ] estatura = {1.73, 1.67, 1.56}; //Array de 3 elementos
```

Para cadenas

```
String[ ] nombre = {"María", "Gerson"}; //Array de 2 elementos
```

Para caracteres

```
char[ ] sexo = {'m', 'f'}; //Array de 2 elementos
```

