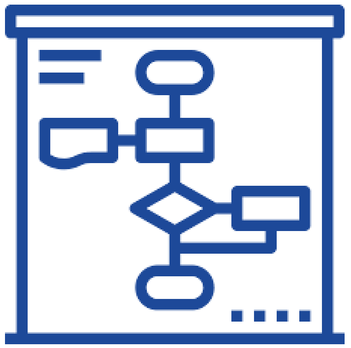
INTRODUCCIÓN A LA PROGRAMACIÓN



OTOÑO, 2021

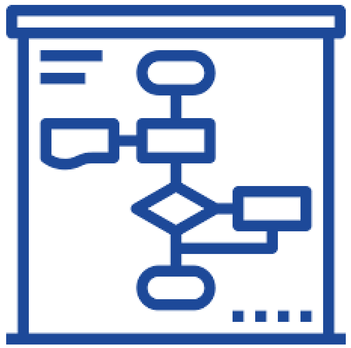
UNIVERSIDAD TECNOLÓGICA DE CHILE
INSTITUTO PROFESIONAL
CENTRO DE FORMACIÓN TÉCNICA
nacap

Actividad a revisar

La gran vida de un programador (software)

https://www.youtube.com/watch?v=Cgc0kt5vecU

Conceptos claves.
Ideas que se deprenden del video.
Ideas relacionadas con tus experiencias previas.
Personaje que te llamó la atención.
Qué aprendiste.



UNIDAD I

Estructuras de control en DFD

Conceptos, simbologías y tipos de datos

Conceptos generales

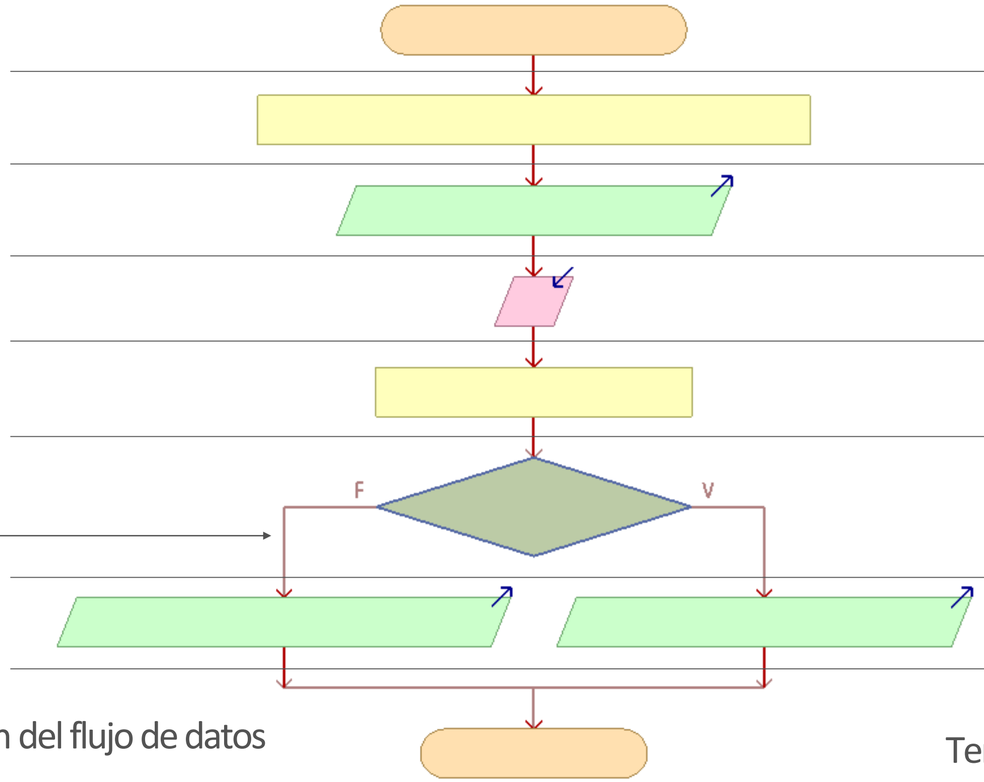
¿cómo es representado un algoritmo?

Descargar software para para realizar los DFD

draw.io

Un **diagrama de flujo de datos (DFD)** es una representación gráfica que utiliza símbolos (cajas) estándar y que tiene los pasos del algoritmo escritos en esas cajas unidas por flechas, denominadas líneas de flujo, que indican la secuencia lógica que debe ejecutar el algoritmo.

Pseudocódigo
lenguaje de especificación (descriptivo y cercano) de algoritmos.



Terminal. Marca el inicio y/o el final en la ejecución de un DFD

Asignación: definir variables, almacenar valores

Escritura o salida: permite mostrar salida de resultados

Lectura o entrada: permite ingresar información desde teclado

Proceso: operaciones (resultados) con los datos ingresados

Estructura de control

Escritura o salida: permite mostrar salida de resultados

Indicadores de la dirección del flujo de datos

Terminal. Marca el final en la ejecución de un DFD

nacap

Tipos de datos

Un tipo de datos es la propiedad de un valor que determina su dominio (qué valores puede tomar), qué operaciones se le pueden aplicar y cómo es representado internamente por el computador.

Caracter

Un *dato* de *tipo de dato* **Caracter** puede tomar como valor cualquier carácter perteneciente a los caracteres que puede representar el ordenador.

Ejemplo: op='$'

Punto flotante (Real)

Un *dato* de *tipo de dato* **Real** puede tomar por valor números decimales, los cuales pertenecen al conjunto de números reales (R).

Ejemplo: 10.7

Entero

Un *dato* de *tipo de dato* **Entero** puede tomar cualquier valor numérico perteneciente al conjunto de los números Z.

Este tipo de dato no puede contener valores decimales.

Ejemplo: -1, 3, 45

Cadena

Un *dato* de *tipo de dato* Cadena puede tomar como valor una secuencia de caracteres (palabras o conjunto de letras).

Ejemplo: "2", "usuario@empresa.cl"

Valores Booleanos (Lógico)

Un *dato* de *tipo de dato* **Lógico** solo puede tomar 2 valores (verdadero o falso).

Un dato de tipo lógico siempre está asociado a que algo se cumpla o no se cumpla. *Ejemplo: sw=falso*

Los tipos de datos son utilizados al momento de definir una variable y ésta debe ser definida antes de usarse.

Operadores y Funciones

Partamos por los más básicos, que son los **operadores matemáticos**

En general son la base del los lenguajes de programación y fueron la razón de su origen

A medida que la tecnología avanzaba se requerían cada vez cálculos más complejos.

La jerarquía de los operadores es igual a la del álgebra, aunque puede alterarse mediante el uso de paréntesis.

Para obtener la raíz cuadra puede utilizar:
*número**(1/2)*
[más adelante lo realizaremos por función]

+	Suma los valores de la izquierda y la derecha

```
(1001+817)
```

−	Resta el valor de la derecha al valor de la izquierda

```
(7-4)
```

*	Multiplica el valor de la izquierda por el de la derecha

```
(5*8)
```

/	Divide el valor de la izquierda por el de la derecha

```
(12/4)
```

**	Calcula la potencia. El valor de la izquierda es la base y el de la derecha es el exponente.

```
(2**5)
```

//	Divide el valor de la izquierda por el de la derecha, y entrega el valor entero del resultado.

```
(5//4)
```

%	Divide el valor de la izquierda por el de la derecha. No entrega el resultado sino que el resto de esta división.

```
(7%5)
```

Operadores y Funciones (consideraciones)

Cómo redondear un número en Python

Redondear un número en Python utilizando el método estándar matemático, es decir:

- De 0.1 a 0.4 decimal, se redondea al entero inferior
- De 0.5 a 0.9 decimal, se redondea al entero superior

```
round(3.2)
[Out] 3

round(3.8)
[Out] 4
```

Cómo truncar un número en Python

Truncar un número en Python. Para truncar un número podemos hacerlo de dos formas, utilizando la función del módulo **math** (***math.trun***) o bien utilizando la función predefinida ***int***

```
int(3.9)
[Out] 3

math.trunc(3.9)
[Out] 3

int(-3.9)
[Out] -3

math.trunc(-3.9)
[Out] -3
```

Ejemplo

Se necesita crear un algoritmo en DFD que tras ingresar una medida expresada en centímetros la convierta en pulgadas.
(1 pulgada = 2.54 cm)

Utilice el valor de la pulgada como constante.

Ejemplo

Se necesita crear un algoritmo en DFD que lea un entero positivo introducido por el usuario y después muestre en pantalla la suma de todos los enteros desde 1 hasta n.
La suma de los n primeros enteros positivos puede ser calculada de la siguiente forma:

$$\text{suma} = \frac{n(n+1)}{2}$$

Ejemplo

Se necesita crear un algoritmo en DFD que permita ingresar el número de partidos ganados y perdidos, por algún equipo en torneo. Se debe mostrar el puntaje total, teniendo en cuenta que por cada partido ganado obtendrá 3 puntos, empatado 1 y perdido 0 puntos.

Ejemplo

Se necesita crear un algoritmo en DFD que permita calcular el área y el perímetro de un triángulo conociendo el largo de los 3 lados.

las fórmulas a aplicar serían:

$$P = L1 + L2 + L3$$

$$S = P / 2$$

$$A = \sqrt{S \times (S - L1) \times (S - L2) \times (S - L3)}$$

L1=4
L2=4
L3=5

Perímetro=12
Área=7.806247498

Operadores lógicos de comparación

igual	==	Compara el valor de la derecha y la izquierda	(10==9)
distinto	!=	Compara el valor de la derecha y la izquierda	(10!=9)
menor que	<	Menor que	(11<7)
mayor que	>	Mayor que	(7>7)
mayor o igual que	>=	Mayor o igual que	(7>=7)
menor o igual que	<=	Menor o igual que	(7<=7)

Los operadores lógicos son usados para operaciones de álgebra Booleana, es decir, para describir relaciones lógicas, expresadas como verdadero o falso.

Operadores lógicos binarios

Símbolo	Significado	Descripción	Ejemplo
AND	Y (Conjunción)	Devuelve verdadero solo cuando ambas proposiciones son verdaderas	(1 = 1) AND (2*2 = 4) es Verdadero
OR	O (Disyunción)	Devuelve verdadero cuando al menos una de las proposiciones es verdadera.	(2=2) OR (2*2=7) es Verdadero

NOT
(negación)

El operador lógico `not (!)` sirve para poder "cambiar" el valor de una operación lógica. Es decir, si uno le aplica este operador a una operación lógica, cambia el valor de VERDADERO a FALSO o viceversa.

not(3>=2); !1=1; (not1!=1)

Tablas de verdad

Para evaluar una expresión lógica se requieren conocer las tablas de verdad de la lógica de Boole.

TABLA DEL "AND" (Y)

AND (Conjunción)

A	B	$A \wedge B$
V	V	V
V	F	F
F	V	F
F	F	F

Si existe una proposición Falsa, la respuesta es Falsa

TABLA DEL "OR" (O)

OR (Disyunción)

A	B	$A \vee B$
V	V	V
V	F	V
F	V	V
F	F	F

Si existe una proposición Verdadera, la respuesta es Verdadera

TABLA DEL "NOT" (NO)

NO (Negación)

A	$\sim A$
V	F
F	V

Devuelve el valor contrario de verdad del enunciado

O Y

!((5>4 || 3<6) && (8=5))

Ejemplos

b>c and d=a

#	a	b	c	d
1	3	10	5	3
2	5	5	9	5

(7>=7) or (119>119)

Ejemplos

Si a = 10, b = 12, c = 13, d =10

((a > b) or (a < c)) and ((a = c) or (a > = b))

6!=6 or 2*4=8

!6=6 and !2>8

not(a=c) and (c>b)

Ejemplos

Según el enunciado diseñe el predicado lógico que lo acepte o rechace.

El número debe ser positivo y mayor que 100

i.

ii.

inacap.cl