CENTRE ESPECÍFIC D'EDUCACIÓ A DISTÀNCIA DE LA COMUNITAT VALENCIANA

Programació

DAW

Unitat 2.

Diagrames de Flux

Autors: Carlos Cacho y Raquel Torres

Revisat per: Lionel Tarazon - lionel.tarazon@ceedcv.es

Fco. Javier Valero – franciscojavier.valero@ceedcv.es José

Manuel Martí - josemanuel.marti@ceedcv.es

Jose Cantó Alonso – j.cantoalonso@edu.gva.es

Llicència

Nomenclatura

Al llarg d'aquest tema s'utilitzaran diferents símbols per a distingir elements importants dins del contingut. Aquests símbols són:

 Important

 Atenció

 Interessant

Contingut

1. Introducció

Com ja vam veure en la unitat anterior, els diagrames de flux (o ordinogrames) són una forma gràfica de representar algoritmes. Dit d'una altra manera, representa les instruccions pas a pas que s'executarien en un programa d'ordinador.

Són molt útils per a practicar i aprendre a pensar de manera algorítmica abans de programar utilitzant llenguatges de programació reals com Java, Python, etc. Els ordinogrames poden crear-se directament amb paper i llapis o utilitzar algun programari com a DIA, yEd, draw.io o Dibuixos de Google.

Vegem un a un els diferents elements d'un ordinograma i com utilitzar-los.

2. Instruccions d'inici i fi

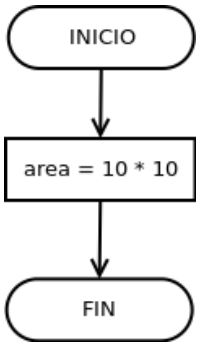
Tots els programes tenen un punt inicial i un punt final que marquen quan comença i acaba un programa. En un ordinograma aquests es representa mitjançant **INICI** i **FI**.

3. Instruccions de processament d'informació

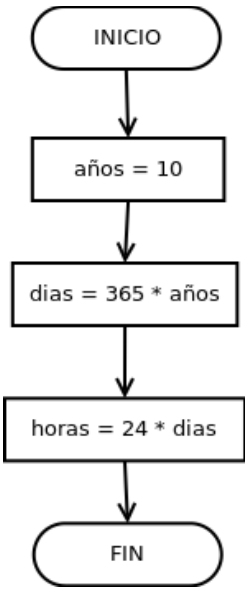
Tots els programes informàtics processen informació, és a dir, realitzen operacions matemàtiques per a calcular, manipular o modificar informació. Com ja vam veure en la unitat anterior les operacions poden ser aritmètiques (+ - * / %), relacionals (< <= > >= == <>) o lògiques (NOT, AND, OR). Al seu torn, s'utilitzen variables (A, B, Edat, Preu, etc.) per a emmagatzemar la informació que estem processant.

L'encarregat de realitzar aquest tipus d'operacions és el **processador d'un ordinador**. En un ordinograma es representen utilitzant rectangles que contenen les instruccions.

Per exemple, el següent ordinograma calcula l'àrea d'un quadrat de costat 10:

Ordinograma	*Explicació*
	Calcula 10*10 i guarda el resultat (100) en la variable "area"

Podem utilitzar tantes instruccions de processament d'informació com necessitem. Per exemple, el següent ordinograma calcula el nombre d'hores que hi ha en 10 anys.

Ordinograma	*Explicació*
	Guarda 10 en la variable "años" Calcula 365 * "años" (365* 10) i guarda el resultat (3650) en la variable "dias" Calcula 24 * "dias" (24 * 3650) i guarda el resultat (87600) en la variable "horas"

Si es desitja, és possible posar diverses instruccions dins d'un mateix rectangle (per a simplificar):


INICIO
años = 10
dias = 365 * años
horas = 24 * dias
FIN

4. Instruccions d'entrada i eixida d'informació

Tots els programes informàtics utilitzen algun tipus d'entrada i eixida d'informació. És una part molt important de la programació ja que, entre altres coses, és el que permet a un usuari interactuar amb un programa. Per exemple:

1. L'usuari introdueix informació per teclat (entrada).
2. El programa processa la informació (fa algun càlcul).
3. El programa mostra el resultat per pantalla (eixida).

Tots els programes d'ordinador, apps de telèfon, pàgines web, etc, segueixen aquests tres passos. Tu prems un botó (o feixos clic, toques la pantalla...), després el processador el processa i finalment succeeix alguna cosa (es visita una pàgina web, s'escolta una cançó, s'envia un whatsapp, etc).

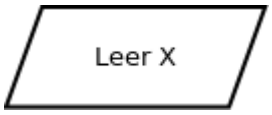
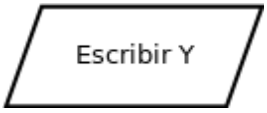
No tindria sentit crear programes sense entrada ni eixida ja que només realitzarien càlculs sense comunicar-se amb l'exterior (com en els dos exemples de l'apartat anterior).

Alguns exemples de dispositius utilitzats per a entrada i eixida d'informació:

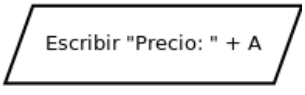
- **Dispositius d'Entrada**: teclat, ratolí, micròfon, escàner, *gps, wifi, etc.
- **Dispositius d'Eixida**: pantalla, altaveus, impressora, wifi, etc.

Ara com ara ens centrarem en l'entrada i eixida més senzilla: el teclat i la pantalla.

En els ordinogrames l'entrada i eixida d'informació es representa mitjançant **un paral·lelogram.**

ENTRADA d'informació	*EIXIDA d'informació*
	
L'usuari introdueix informació per teclat i es guarda en la variable X	Mostra per pantalla el contingut de la variable Y

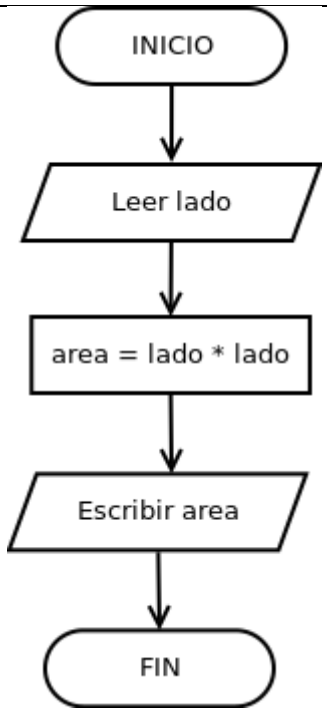
La instrucció d'EIXIDA d'informació és molt versàtil. També pot utilitzar-se per a mostrar text, a més de combinacions de text i variables.

Ordinograma	*Explicació*
	Mostra per pantalla el text "Hola mundo!"
	Mostra per pantalla el text "Precio: " seguit del valor de la variable A. Si per exemple A val 15, mostrarà el text "Precio: 15".

Recordes el primer ordinograma de l'apartat 3? Simplement calculava l'àrea d'un quadrat de costat 5 i ni si tan sols mostrava el resultat per pantalla.

El modificarem afegint entrada i eixida de manera que primer li demane a l'usuari que introduïsca el costat del quadrat, després calcule l'àrea, i finalment la mostre.

Ordinograma	*Explicació*



L'usuari introduïx un número per teclat (entrada) i es guarda en la variable "lado"

Calcula "lado" * "lado" i guarda el resultat en la variable "area"

Mostra per pantalla (eixida) el contingut de la variable "area"

Si creàrem un programa seguint aquest algoritme millorat (amb entrada i eixida) **ens permetria utilitzar-lo per a calcular l'àrea de qualsevol quadrat.** Això és molt més útil i general que el programa original que només calculava l'àrea un quadrat de costat 5.

Per això, afegir entrada i eixida als programes és fonamental perquè siguen d'utilitat.

Ara modificarem l'ordinograma que calculava els dies i hores per a **afegir-li entrada i eixida**. Volem que el programa li demane a l'usuari que introduïsca per teclat els anys, calcule els dies i les hores, i ho mostre per pantalla.

Ordinograma *Explicació*

L'usuari introduïx un número per teclat (entrada) i es guarda en la variable "años"

Calcula 365 * "años" i ho guarda a la variable "dias". Calcula 24 * "dies" i ho guarda a la variable "horas"

Mostra per pantalla (eixida) el contigut de la variable "horas"

De nou, si creàrem un programa que seguira els passos d'aquest ordinograma, el podríem utilitzar per a calcular el nombre d'hores dels anys que volguérem, tantes vegades com volguérem.

Ara és un bon moment per a fer els **exercicis de l'1 al 7**.

5. Estructures de control

Fins ara hem vist algoritmes (representats com ordinogrames) en els quals les instruccions s'executen seqüencialment (una després de l'altra). Però sovint és necessari dissenyar algoritmes les instruccions dels quals no s'executen seqüencialment, per al que és necessari utilitzar estructures de control.

Les estructures de control són utilitzades per a controlar la seqüència (l'ordre) en el qual s'executen les instruccions. Existeixen dos tipus:

- **Estructures alternatives**: permeten alternen entre diferents instruccions (executar les unes o les altres) depenent d'una condició. Poden ser simples, dobles o múltiples.
- **Estructures repetitives**: permeten repetir instruccions (executar-les diverses vegades).

En aquesta unitat ens centrarem en les estructures alternatives. En la següent unitat veurem les estructures repetitives.

5.1. Estructures alternatives

Controlen l'execució o la no execució d'una o més instruccions en funció que es complisca una condició. Dit d'una altra manera, **s'utilitzen perquè succeïsquen coses diferents depenent d'una condició**.

Per exemple, en introduir una contrasenya si aquesta és correcta s'iniciarà sessió, però si és incorrecta mostrarà un missatge d'error. Per posar un altre exemple, quan estrenys una lletra o un número del teclat aquesta es mostrarà per pantalla, però si és la tecla d'intro llavors el cursor baixarà a la següent línia.

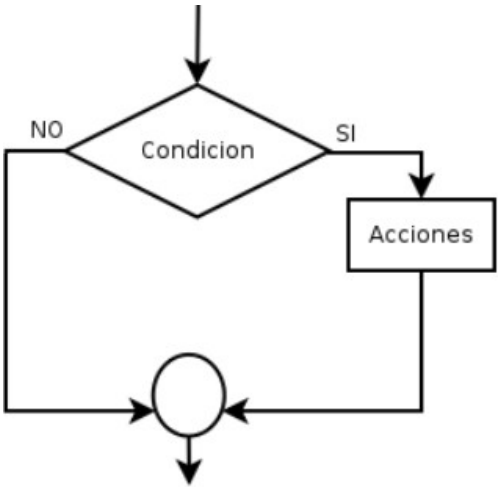
Pot semblar obvi però això succeeix així perquè un programador ha utilitzat una estructura alternativa per a indicar exactament què ha de succeir en cada cas. Les estructures alternatives són molt importants per a establir diferents comportaments a un programa.

Existeixen tres tipus d'estructures alternatives: **Simple**, **Doble** i **Múltiples**.

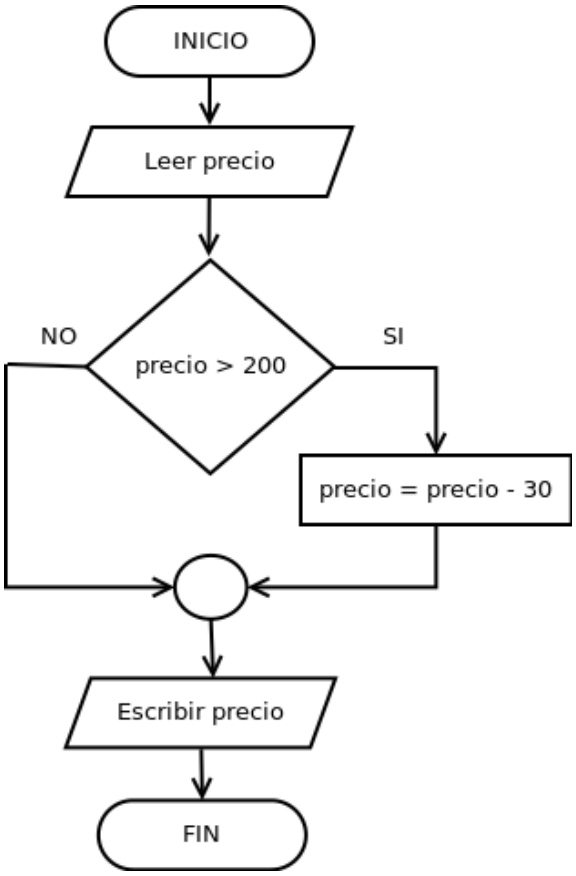
Totes elles utilitzen condicions, com (preu > 200), (edat >= 18), (contrasenya == “1234”), etc. En un ordinograma una condició s'expressa mitjançant un rombe.

5.1.1. Estructura alternativa simple

L'estructura alternativa simple és molt senzilla. Si la condició és vertadera s'executarà una o diverses instruccions concretes, però si és falsa aquestes no s'executaran. Es representa així:

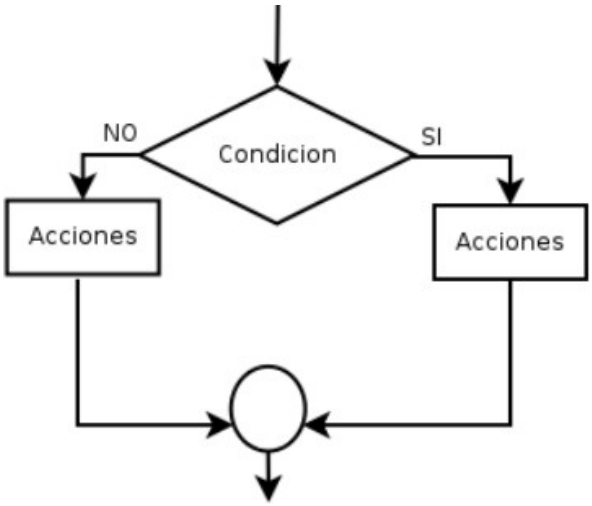


Vegem un exemple senzill. Un programa que llig per teclat el preu inicial d'un producte, si és superior a 200 € se li aplica un descompte de 30 €, i finalment el mostra per pantalla.

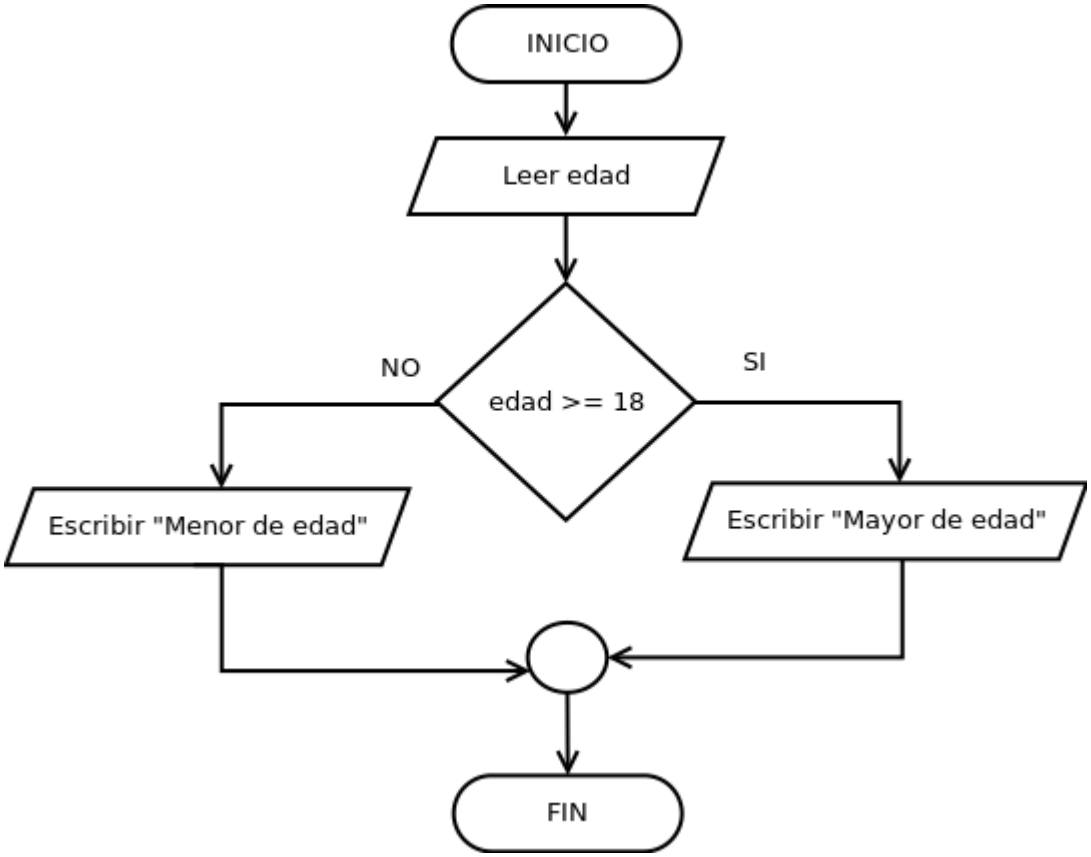


5.1.2. Estructura alternativa doble

L'estructura alternativa doble és molt similar a la simple. L'única diferència és que si la condició és una certa s'executaran unes instruccions i si és falsa s'executaran altres diferents.



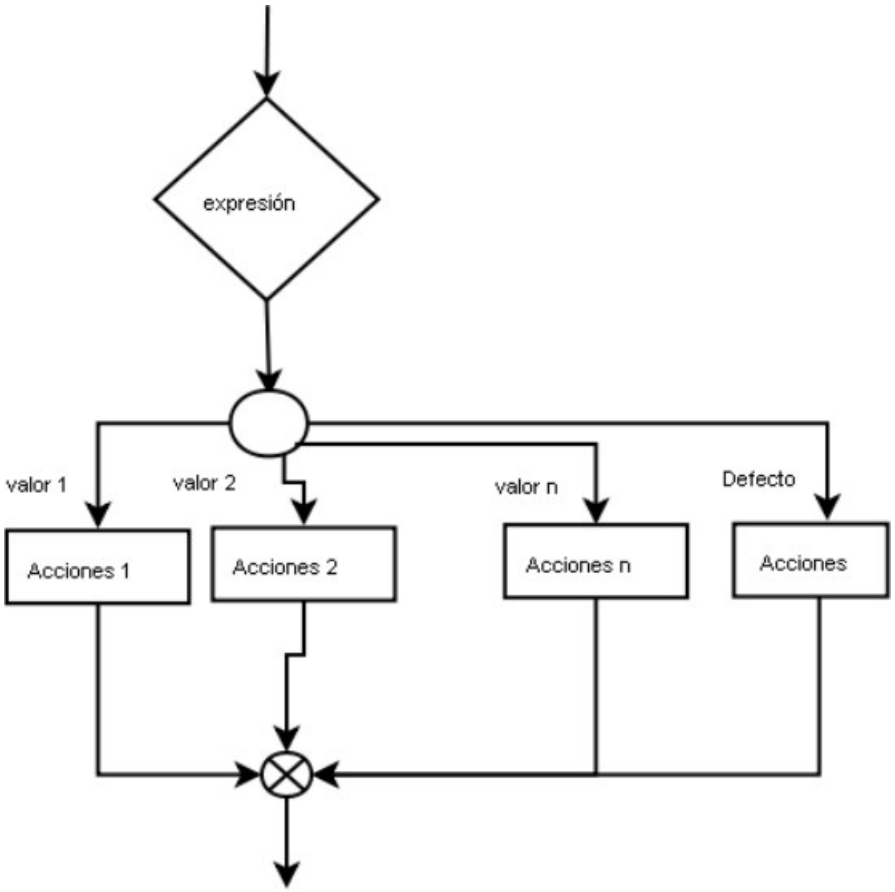
Vegem un exemple. Un programa que demana l'edat per teclat, si és major o igual a 18 mostrarà per pantalla el text “Major d'edat”, en cas contrari mostrarà “Menor d'edat”.



5.1.3. Estructura de alternativa múltiple

En l'estructura alternativa múltiple es poden especificar diferents instruccions depenent del valor d'una expressió. I cal prestar atenció a aquest tret: depèn del valor d'una expressió, i no d'una condició. L'expressió donarà un resultat concret (per exemple 10, -5, 0.25, etc.) i s'executaran les instruccions associades a cada valor.

Si el resultat de l'expressió no és igual a cap dels valors indicats, s'executaran les instruccions per defecte (si s'ha indicat, això és opcional).


expresión
valor 1
valor 2
valor n
Defecto
Acciones 1
Acciones 2
Acciones n
Acciones

6. Agraïments

Anotacions actualitzades i adaptades al CEEDCV a partir de la següent documentació:

[1] Apuntes Programación de José Antonio Díaz-Alejo. IES Camp de Morvedre.