Algorismia i programació BAT_PTD1 8.2

Lloc:	Institut d'Ensenyaments a Distància de les Illes Balears	*Imprès Per:*	PERE MIQUEL ROSSELLO ESPASES
Curs:	Participant 041	*Data:*	dilluns, 15 d'abril 2024, 20:33
Llibre:	Algorismia i programació BAT_PTD1 8.2		

Taula de continguts

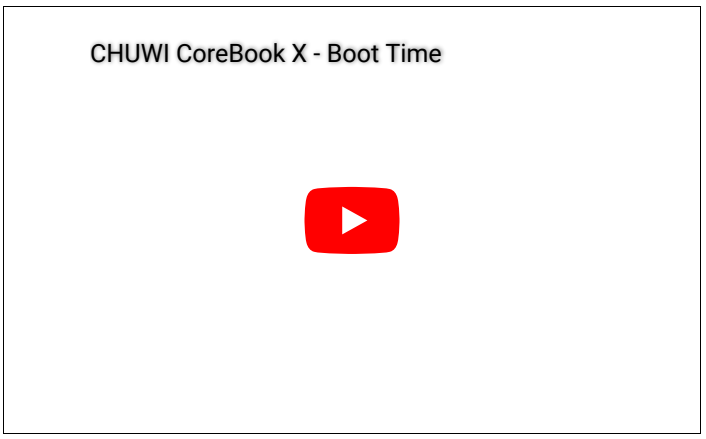
1. Programació d'ordinadors

La programació d'ordinadors és el procés de crear seqüències d'instruccions que una computadora pot entendre i executar per realitzar tasques específiques . Aquestes instruccions s'escriuen en un llenguatge de programació específic i s'anomenen codi font. Els programadors utilitzen aquests llenguatges per desenvolupar software que pot realitzar una àmplia varietat de tasques, des de simples càlculs fins a aplicacions complexes i sistemes informàtics.

El procés de programació sol implicar diverses etapes :

1. **Anàlisi del problema**: Comprendre les necessitats dels usuaris i els requisits del sistema.
2. **Disseny de l'algorisme**: Crear un pla detallat per resoldre el problema utilitzant un algorisme eficient.
3. **Implementació**: Escriure el codi font en el llenguatge de programació seleccionat.
4. **Proves**: Provar el programa per assegurar-se que funcioni com es desitja i corregir els errors (bugs) que es trobin.
5. **Optimització**: Millorar el rendiment i l'eficiència del programa, si és necessari.
6. **Documentació**: Escriure documentació clara per a altres programadors o usuaris del programa.

Els programadors poden treballar en una àmplia varietat de camps i desenvolupar software per a diferents plataformes i dispositius, com ara aplicacions mòbils, pàgines web, sistemes operatius, jocs, sistemes de gestió de bases de dades, i molt més. La programació d'ordinadors requereix una combinació de coneixements tècnics, habilitats de resolució de problemes i creativitat per crear solucions efectives i innovadores.



Vídeo: *CHUWI CoreBook X - Boot Time*

Més informació...

1.1. Algorisme

Un algorisme és un conjunt d'instruccions sequencials i precises que descriuen una solució a un problema específic . Aquests algorismes són la base de la programació i es poden expressar de diverses maneres, incloent el pseudocodi o els diagrames de flux . Un algorisme eficient és aquell que soluciona el problema de manera correcta i òptima, utilitzant els recursos disponibles de la millor manera possible.

Un exemple senzill d'algorisme podria ser un algorisme per trobar la suma dels primers n nombres naturals:

▪ less Copy code

```
1. Inicialitzar la variable suma a 0.2. Inicialitzar la variable n a un nombre
1 natural donat.3. Per a cada nombre i des de 1 fins a n, fer:     a. Sumar i a la
  variable suma.4. Imprimir la suma.
```

Aquest algorisme podria ser implementat en diversos llenguatges de programació com ara Python, JavaScript o Java. Aquí tens un exemple en JavaScript :

▪ javascript Copy code

```
function sumaNaturals(n) {    let suma = 0;    for (let i = 1; i <= n; i++) {
suma += i;    }    return suma;} const n = parseInt(prompt("Introdueix el nombre
1 n:"));console.log("La suma dels primers", n, "nombres naturals és:",
  sumaNaturals(n));
```

Aquest codi segueix l'estructura bàsica de l'algorisme anterior. La funció `sumaNaturals` calcula la suma dels primers n nombres naturals utilitzant un bucle `for`, i després es mostra el resultat utilitzant `console.log`. En aquest cas, es fa servir `prompt` per llegir l'entrada de l'usuari des del navegador web.

Aquest és només un exemple senzill, però els algorismes poden ser molt més complexos i utilitzar diverses tècniques i estructures de control per resoldre problemes més complicats. La capacitat de dissenyar i implementar algorismes eficients és fonamental per als desenvolupadors de programari i els professionals en àmbits relacionats amb la informàtica.

Més informació...

```
SubAlgoritmo Seleccionar_palabra_secreta (Idioma)
FinSubAlgoritmo

SubAlgoritmo Pasar_palabra_a_guiones (Palabra)
FinSubAlgoritmo

Proceso Juego_del_Ahorcado
        Seleccionar_palabra_secreta (Idioma)
        Pasar_palabra_a_guiones (Palabra)
        Repetir
                Leer Introducimos_una_letra
                Si 'Letra en palabra secreta' Entonces
                        Si '¿Hemos ganado?' Entonces
                        FinSi
                SiNo
                        Si '¿Hemos perdido?' Entonces
                        FinSi
                FinSi
        Hasta Que '¿Hemos ganado?' O '¿Hemos perdido?'
FinProceso
```

1.2. Diagrama de flux

Un diagrama de flux és una representació visual d'un algorisme o procés . S'utilitza per il·lustrar les diferents etapes d'un procés i les relacions entre elles mitjançant símbols gràfics i connectors. Aquests diagrames són útils per comunicar de manera clara i concisa el flux d'un procés i són àmpliament utilitzats en diversos camps, incloent la programació, l'enginyeria, la gestió de projectes i altres àrees.

Els elements principals d'un diagrama de flux inclouen :

1. **Inici/Final**: Indica l'inici o el final del procés. S'expressa típicament amb una el·lipse.
2. **Operació/Procediment**: Representa una tasca o una acció que s'executa dins del procés. S'expressa amb un rectangle.
3. **Decisió**: Indica una condició que determina el camí a seguir en el procés. S'expressa amb un rombe o un rectangle amb un símbol de pregunta a dins.
4. **Connector**: Connecta diferents parts del diagrama i indica la continuïtat del flux d'execució. Es representen amb línies que connecten els diversos símbols.

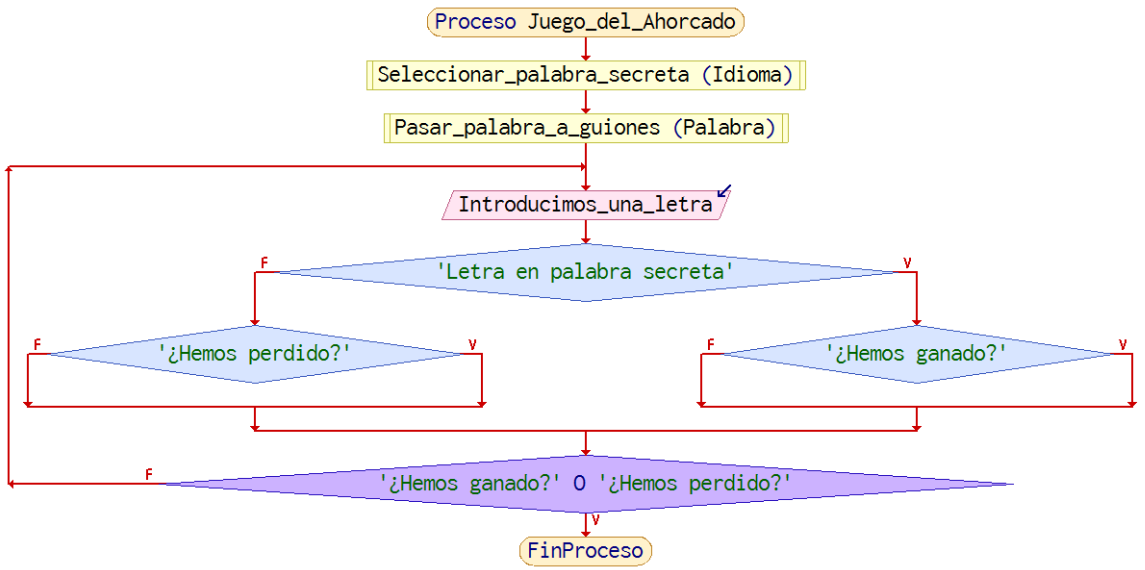
Un exemple simple de diagrama de flux seria :

▪ scss Copy code

```
[Inici] --> [Operació 1] --> [Decisió] --> (Sí) --> [Operació 2] --> [Final]
\                    /                    \--> (No) --------/
```

En aquest exemple, el procés comença amb una operació, després es pren una decisió basada en una condició. Si la condició és certa, es realitza una altra operació i el procés finalitza. Si la condició és falsa, el procés continua amb una altra operació i després finalitza. Això es representa visualment amb els símbols adequats i les línies de connexió.

Més informació...



2. Programació estructurada

La programació estructurada és un paradigma de programació que es basa en l'ús de construccions de control com ara seqüències, iteracions (bucles) i seleccions (condicionals) per a organitzar el flux d'execució del programa . Aquest paradigma promou l'ús de tècniques com ara la descomposició modular i la creació de programes amb una estructura lògica i clara.

Alguns dels principis fonamentals de la programació estructurada són :

1. **Seqüencialitat**: Les instruccions s'executen en seqüència, una després de l'altra.
2. **Selecció**: S'utilitzen estructures condicionals com ara `if-else` per fer decisions basades en certes condicions.
3. **Iteració**: S'utilitzen estructures de bucle com ara `for`, `while` o `do-while` per repetir blocs d'instruccions mentre es compleixin certes condicions.
4. **Abstracció**: Es divideix el programa en funcions o mòduls més petits, cadascun amb una única responsabilitat i que es poden reutilitzar en altres parts del programa.

La programació estructurada és important perquè facilita la comprensió, la depuració i el manteniment del codi, ja que fa que els programes siguin més fàcils de llegir i de gestionar . A més, promou la modularitat i la reutilització de codi, que són aspectes fonamentals en el desenvolupament de programari.

Més informació...

```
1
/*  * L'abecedari català + castellà es composa  * de les vint-i-vuit lletres següents:  * a, b, c, ç, d, e, f, g, h, i, j, k, l, m, (Has encertat!) * n, ñ, o, p, q, r, s, t, u, v, w, x, y, z. (Has fallat!) */if (((lletra >= "a") && (lletra <= "m")) || (lletra == "ç")) { window.alert("Has encertat!");        // Afegim lletra a Paraula i actualitzam la pantalla. Paraula = Paraula + lletra + " ";    document.getElementById("Paraula").innerHTML = Paraula;} else if (((lletra >= "n") && (lletra <= "z")) || (lletra == "ñ")) {    window.alert("Has fallat!");        // Afegim lletra a Lletres i actualitzam la pantalla.    Lletres = Lletres + lletra + " ";    document.getElementById("Lletres").innerHTML = Lletres;     // Decrementam el comptador Vides.        Vides = Vides - 1;    // window.alert(Vides);} else { window.alert("Caràcter incorrecte!");} // Actualitzam Vides a la pantalla.document.getElementById("Vides").innerHTML = "&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;" + Vides; // Comprovam si has perdut! o has guanyat!.if (Vides <= 0) {    window.alert("i has perdut!");    AturaTot();} else {    if (Paraula.length >= 14) {        window.alert("i has guanyat!");        AturaTot();    }}
```

2.1. Estructures de control

1. **Estructura de selecció (condicionals)**:

- javascript Copy code

```javascript
if (condicio) {
    // Bloc d'instruccions si la condició és certa
} else {
    // Bloc d'instruccions si la condició és falsa
}
```

2. **Estructura d'iteració (bucles)**:

- javascript Copy code

```javascript
for (let i = 0; i < 5; i++) {
    // Bloc d'instruccions per a cada iteració
}

let x = 0;
while (x < 5) {
    // Bloc d'instruccions mentre la condició sigui certa
    x++;
}
```

3. **Estructura de transferència de control**:

javascript Copy code

```javascript
// Interrupció del bucle
for (let i = 0; i < 10; i++) {
    if (i === 5) {
        break;
    }
}

// Salta a la següent iteració del bucle
for (let i = 0; i < 5; i++) {
    if (i === 2) {
        continue;
    }
    console.log(i); // Imprimirà 0, 1, 3, 4
}

// Retorna un valor des d'una funció i finalitza l'execució de la funció
function suma(a, b) {
    return a + b;
}
```

Aquests són exemples de les estructures de control més comuns en JavaScript, que es poden utilitzar per gestionar el flux d'execució dels programes i prendre decisions dinàmiques en funció de les condicions.

Més informació...

TASCA D'AMPLIACIÓ:

Et propòs que realitzis aquesta tasca: Qüestionari d'ampliació BAT_PTD1 1.1, que trobaràs en la capsa del lliurament 8 per practicar els continguts vists fins ara.

3. Llenguatge de programació

El llenguatge de programació és un conjunt de regles i símbols que s'utilitzen per escriure instruccions que una computadora pot entendre i executar . Hi ha molts llenguatges de programació diferents, cadascun amb les seves pròpies característiques i usos. Alguns exemples de llenguatges de programació populars inclouen Python, Java, C++, JavaScript, Ruby i moltíssims més. Cada llenguatge de programació té les seves pròpies avantatges i desavantatges, i la seva elecció depèn sovint dels requisits del projecte, la preferència personal i altres factors com la facilitat d'aprenentatge i la disponibilitat de llibreries o suport comunitari.

Més informació...

Prototip Joc del Penjat

Escriu una lletra minúscula Comprovar

Paraula

_ _ _ _ _ _ _

Vides

-

Lletres

_ _ _ _ _ _ _

Crèdits: El joc del penjat on Scratch

3.1. Llenguatge de programació visual

El llenguatge de programació visual és una forma de codificar que utilitza elements gràfics com a nodes, icones o blocs interconnectats per crear algorismes o aplicacions . Aquest tipus de llenguatge sol ser molt intuïtiu i accessible per a principiants i per a aquells que prefereixen una interfície més visual en lloc de codi de text. Alguns exemples populars inclouen Scratch, Blockly i LabVIEW . Aquests llenguatges són freqüentment utilitzats en l'ensenyament de la programació als nens, però també són utilitzats en entorns professionals per a tasques com el desenvolupament de software, el control de processos industrials i la creació de simulacions.

FRAGMENT DE TEXT:

Llista de llenguatges visuals

Es pot fer esment dels més coneguts :

Programaris educatius

- App Inventor del MIT: creador d'aplicacions per Android. Basat en el programari Blockly.
- Snap!: llenguatge d'edició educatiu.

Programaris multimèdia[modifica]

- Blender: editor 3D.
- Max (programari): adreçat al camp musical.
- Pure Data: adreçat al camp musical i també enllumenat.

Programaris video jocs[modifica]

- Unity: entorn creació de jocs.
- Unreal Engine: motor de creació de jocs.

Programaris de simulació de sistemes[modifica]

- LabVIEW: adreçat a enginyers i científics.
- GNU Radio: per a definir ràdios via programari.
- Simulink: simulació de sistemes.

Més informació...

TASQUES D'APRENENTATGE:

Et propòs que realitzis aquestes tasques: Joc de frontó en Scratch BAT_PTD1, Joc d'obstacles en Scratch BAT_PTD1 i Joc d'asteroides en Scratch BAT_PTD1, que trobaràs en la capsa del lliurament 8 per practicar els continguts vists fins ara.