Projectes de desenvolupament d'aplicacions. Entorn integrats de desenvolupament.

Per poder elaborar una aplicació és necessari utilitzar una sèrie d'eines que ens permetin escriure-la, depurar-la, traduir-la i executar-la. Aquest conjunt d'eines es coneix com a entorn de desenvolupament integrat i la seva funció és proporcionar un marc de treball per al llenguatge de programació.

Un entorn de desenvolupament integrat, o IDE, és un programa compost per una sèrie d'eines que utilitzen els programadors per desenvolupar codi. Aquest programa pot estar pensat per a la seva utilització amb un únic llenguatge de programació o bé pot donar cabuda a diversos.

Interfícies gràfiques d'usuari
Les interfícies gràfiques d'usuari són un conjunt de funcionalitats que permeten incorporar, editar i eliminar components gràfics de forma senzilla en l'aplicació que s'està desenvolupant. Aquests components facilitaran la interacció de l'usuari amb l'ordinador.

Editor de text
Un editor de text és un programa que permet crear i modificar arxius digitals compostos únicament per text sense format, coneguts comunament com arxius de text o text pla.

L'editor de text és l'eina més utilitzada perquè ofereix la possibilitat de crear i modificar els continguts desenvolupats, el codi de programació que farà funcionar adequadament l'aplicació.

Sempre que l'IDE reconegui el llenguatge de programació (o disposi del component adequat) l'editor oferirà:

Ressaltat de sintaxi (syntax highlighting): les paraules clau seran reconegudes amb colors, cosa que facilitarà molt la feina del programador.

Compleció de codi (code completion): es reconeixerà el codi que s'està escrivint i, per exemple en un objecte o classe, oferirà els seus atributs, propietats o mètodes per tal que el programador seleccioni quin vol referenciar.

Corrector d'errors, normalment des d'un punt de vista de sintaxi.

Regions plegables: ocultació de certes parts del codi, per tal de facilitar la visualització d'aquells fragments més rellevants.
Compilador

En funció del llenguatge de programació utilitzat, l'IDE podrà oferir la funcionalitat de compilar-lo. Un compilador tradueix un codi de programació en un llenguatge màquina capaç de ser interpretat pels processadors i de ser executat. A l'hora de compilar un codi de programació, els entorns integrats de desenvolupament disposaran de diferents fases d'anàlisi del codi, com són la fase semàntica i la fase lexicogràfica. La compilació mostrarà els errors trobats o generarà codi executable, en el cas de trobar-ne cap.

Intèrpret
L'intèrpret tradueix el codi d'alt nivell a codi de bytes, però, a diferència del compilador, ho fa en temps d'execució.

Depurador
El depurador és un programa que permet provar i depurar el codi font d'un programa, facilitant la detecció d'errors.

Algunes de les funcionalitats típiques dels depuradors són:

Permetre l'execució línia a línia del codi validant els valors que van adquirint les variables.
Pausar el programa en una determinada línia de codi, fent ús d'un o diversos punts de ruptura (breakpoints).
Alguns depuradors ofereixen la possibilitat de poder modificar el contingut d'alguna variable mentre s'està executant.
Opcionalment, poden presentar un sistema d'accés a bases de dades i gestió d'arxius, un sistema de control de versions, un sistema de proves, un sistema de refactorització i un generador automàtic de documentació.

Accés a bases de dades i gestió d'arxius
Els llenguatges de quarta generació ofereixen la possibilitat d'integrar codi d'accés a bases de dades (o codi de sentències estructurades). Per facilitar aquesta funcionalitat és molt recomanable disposar de la possibilitat d'accedir directament a la base de dades des del mateix entorn de desenvolupament i no haver-ne d'utilitzar un altre. El mateix succeirà amb la gestió d'arxius.

Control de versions
A mesura que un programador va desenvolupant noves línies de codi, és important tenir un històric de la feina feta o de les versions que s'han donat per bones o s'han modificat. El control de versions permet tornar a una versió anterior que sigui estable o que compleixi unes determinades característiques que els canvis fets no compleixen.

Refactorització
La refactorització (refactoring) és una tècnica de l'enginyeria de programari dirigida a reestructurar un codi font, alterant la seva estructura interna sense canviar el seu comportament extern.

Documentació i ajuda
La documentació i ajuda proveeix accés a documentació, manuals i ajuda contextual sobre les instruccions i la sintaxi dels llenguatges suportats.

Comentaris en el codi

Els comentaris en Java i en qualsevol llenguatge de programació d'una eina que serveix per recolzar la documentació dels programes que desenvolupem i així facilitar la seva comprensió posterior per part d'alguna altra persona que comprengui alguna cosa de Java o el llenguatge en particular.

Poden ser:
d'una sola linea si posem una doble barra (// comentari)
de mes d'una linea si els tanquem amb asterisc barra
/*
comentari
*/

i de decumentació que s'inicien amb /** i es tanquen amb /*

els de linea i multiple lienea es solen utilitzar per explicar cparts concretes del codi mentre que els de documentació per explicar seccions mes amplies

Fonaments de programació.

Què és un programa?

Un primer pas per poder començar a estudiar com cal fer un programa informàtic és tenir clar què és un programa. En contrast amb altres termes usats en informàtica, és possible referir-se a un "programa" en el llenguatge col·loquial sense haver d'estar parlant necessàriament d'ordinadors. Us podríeu estar referint al programa d'un cicle de conferències o de cinema. Però, tot i no tractar-se d'un context informàtic, aquest ús ja us aporta una idea general del seu significat: un conjunt d'esdeveniments ordenats de manera que succeeixen de forma seqüencial en el temps, un darrere l'altre.

Un altre ús habitual, ara ja sí que vinculat al context de les màquines i els autòmats, podria ser per referir-se al programa d'una rentadora o d'un robot de cuina. En aquest cas, però, el que succeeix és un conjunt no tant d'esdeveniments, sinó d'ordres que l'electrodomèstic segueix ordenadament. Un cop seleccionat el programa que volem, l'electrodomèstic fa totes les tasques corresponents de manera autònoma.

Per exemple, el programa d'un robot de cuina per fer una crema de blat de moro seria:

Espera que introduïu blat de moro i mantega.

Gira durant un minut, avançant progressivament de la velocitat 1 a la 5.

Espera que introduïu llet i sal.

Gira durant 30 segons a velocitat 7.

Gira durant 10 minuts a velocitat 3 mentre cou a una temperatura de 90 graus.

S'atura. La crema està llesta!

Aquest conjunt d'ordres no és arbitrari, sinó que serveix per dur a terme una tasca de certa complexitat que no es pot fer d'un sol cop. S'ha de fer pas per pas. Totes les ordres estan vinculades entre si per arribar a assolir aquest objectiu i, sobretot, és molt important la disposició en què es duen a terme.

Entrant ja, ara sí, en el món dels ordinadors, la manera com s'estructura la mena de tasques que aquests poden fer té molt en comú amb els programes d'electrodomèstics. En aquest cas, però, en lloc de transformar ingredients (o rentar roba bruta, si es tractés d'una rentadora), el que l'ordinador transforma és informació o dades. Un programa informàtic no és més que un seguit d'ordres que es porten a terme seqüencialment, aplicades sobre un conjunt de dades.

Quines dades processa un programa informàtic? Bé, això dependrà del tipus de programa:

Un editor processa les dades d'un document de text.

Un full de càlcul processa dades numèriques.

Un videojoc processa les dades que diuen la forma i ubicació d'enemics i jugadors, etc.

Un navegador web processa les ordres de l'usuari i les dades que rep des d'un servidor a Internet.

Per tant, la tasca d'un programador informàtic és escollir quines ordres constituiran un programa d'ordinador, en quin ordre s'han de dur a terme i sobre quines dades cal aplicar-les, perquè el programa porti a terme la tasca que ha de resoldre. La dificultat de tot plegat serà més gran o petita depenent de la complexitat mateixa d'allò que cal que el programa faci. No és el mateix establir què ha de fer l'ordinador per resoldre una multiplicació de tres nombres que per processar textos o visualitzar pàgines a Internet.

Executar un programa

Per “executar un programa” s’entén fer que l’ordinador segueixi totes les seves ordres, des de la primera fins la darrera.

D’altra banda, un cop fet el programa, cada cop que l’executeu, l’ordinador complirà totes les ordres del programa.

De fet, un ordinador és incapaç de fer absolutament res per si mateix, sempre cal dir-li què ha de fer. I això se li diu mitjançant l’execució de programes. Tot i que des del punt de vista de l’usuari pot semblar que quan es posa en marxa un ordinador aquest funciona sense executar cap programa concret, cal tenir en compte que el seu sistema operatiu és un programa que està sempre en execució.

Tipus d'ordres que accepta un ordinador

Per dur a terme la tasca encomanada, un ordinador pot acceptar diferents tipus d’ordres. Aquestes es troben limitades a les capacitats dels components que el conformen, de la mateixa manera que el programa d’una rentadora no pot incloure l’ordre de gratinar, ja que no té gratinador. Per tant, és important tenir present aquest fet per saber què es pot demanar a l’ordinador quan creeu un programa.

El processador és el centre neuràlgic de l’ordinador i l’element que és capaç de dur a terme les ordres de manipulació i transformació de les dades. Un conjunt de dades es pot transformar de moltes maneres, segons les capacitats que ofereixi cada processador. Tot i així, hi ha moltes transformacions que tots poden fer. Un exemple és la realització d’operacions aritmètiques (suma, resta, multiplicació, divisió), tal com fan les calculadores.

La memòria permet emmagatzemar dades mentre aquestes no estan essent directament manipulades pel processador. Qualsevol dada que ha de ser tractada per un programa estarà a la memòria. Mitjançant el programa es pot ordenar al processador que desi certes dades o que les recuperi en qualsevol moment. Normalment, quan es parla de memòria a aquest nivell ens referim a memòria dinàmica o RAM (random access memory, memòria d’accés aleatori). Aquesta memòria no és persistent i un cop acaba l’execució del programa totes les dades amb les quals tractava s’esvaeixen. Per tant, la informació no es desarà entre successives execucions diferents d’un mateix programa.

En certs contextos és possible que ens trobem també amb memòria ROM (read-only memory, memòria només de lectura). En aquesta, les dades estan

predefinides de fàbrica i no s'hi pot emmagatzemar res, només podem llegir el que conté. Cal dir que no és el cas més habitual.

El sistema d'entrada/sortida (abreujat com a E/S) permet l'intercanvi de dades amb l'exterior de l'ordinador, més enllà del processador i la memòria. Això permet traduir la informació processada en accions de control sobre qualsevol perifèric connectat a l'ordinador. Un exemple típic és establir una via de diàleg amb l'usuari, ja sigui per mitjà del teclat o del ratolí per demanar-li informació, com per la pantalla, per mostrar els resultats del programa. Aquest sistema és clau per convertir un ordinador en una eina de propòsit general, ja que el capacita per controlar tota mena d'aparells dissenyats per connectar-s'hi.

Disseny d'algorismes.

És una seqüència ordenada de passos que descriu el procés que ha de seguir, per donar solució a un problema específic.
Les principals estructures son:
• Composició secuencial
- les instruccions s'executen en secuencia, una rere l'altra
• Composició alternativa o condicional
- en funció d'una condició es trien unes instruccions o altres
• Estructura iterativa o bucle o lazo
- es repeteixen unes instruccions mentre es compleixen una condició
• Estructura recursiva
- es repeteixen unes instruccions mitjançant un mètode que s'invoca a sí mateix
Les instruccions d'un programa Java poden ser:
• simples:
- expressions: d'assignació, increment o decrement
- anomenades a mètodes
- creació d'objectes
- instruccions de control: if, switch, while, do-while, for
• compostes:
- es tanquen entre claus {}, i també es diuen blocs
- poden contenir moltes instruccions i declaracions;
- les declaracions del bloc només són visibles en ell, i en els blocs continguts en ell

Prova de programes. Depuració d'errors.

Objectiu: eliminació del errors (s'estima que s'ocupa el 50% del recursos en aquesta tasca).
Convé començar per diferenciar les tècniques de verificació formal (proving) de les de prova (testing). Amb les primeres es demostra formalment que el programa satisfà l'especificació. Amb les segones es tracta de fer-ne proves executant el programa amb diferents conjunts de dades, el més completes que es pugui, per comprovar la correctesa del programa.

pots utilitzar ferramentes com visual studio

Documentació dels programes.

les parts que ha de tenir son:
Nom de la classe, descripció general, número de versió, nom d'autors.
Documentació de cada constructor o mètode (especialment els públics) incloent: nom del constructor o mètode, tipus de retorn, noms i tipus de paràmetres si n'hi ha, descripció general, descripció de paràmetres (si n'hi ha), descripció del valor que retorna.
es documenta a partir de tags que diuen allò que s'esta documentant

En la tabla siguiente mostramos algunas de las palabras reservadas (tags), aunque hay más de las que aquí incluimos. Si necesitas una lista completa de las tags con su correspondiente uso realiza una búsqueda de la cadena "javadoc tags" en bing, google o yahoo.

TAG	DESCRIPCIÓN	COMPRENDE
@author	Nombre del desarrollador.	Nombre autor o autores
@deprecated	Indica que el método o clase es obsoleto (propio de versiones anteriores) y que no se recomienda su uso.	Descripción
@param	Definición de un parámetro de un método, es requerido para todos los parámetros del método.	Nombre de parámetro y descripción
@return	Informa de lo que devuelve el método, no se aplica en constructores o métodos "void".	Descripción del valor de retorno
@see	Asocia con otro método o clase.	Referencia cruzada referencia (#método(); clase#método(); paquete.clase; paquete.clase#método()).
@version	Versión del método o clase.	Versión

Entorns de desenvolupament de programes.

es tracta de les apps que s'utilitzen per programar n'hi ha molts i per a molts tipus de llenguatges

Conoce estos 6 entornos de programación (IDE) para programar en varios lenguajes

- NetBeans. Netbeans también es un **entorno** de **programación** muy utilizado por los programadores. ...
- Visual Studio. ...
- JetBrain. ...
- QtCreator. ...
- CodeLite.

BIBLIOGRAFIA

https://ioc.xtec.cat/materials/FP/Recursos/fp_dam_m05_/web/fp_dam_m05_htmlindex/WebContent/u1/a2/continguts.html

https://www.programarya.com/Cursos/Java/Comentarios

https://ioc.xtec.cat/materials/FP/Recursos/fp_asix_m03_/web/fp_asix_m03_htmlindex/WebContent/u1/a1/continguts.html

https://ocw.unican.es/pluginfile.php/293/course/section/228/cap3-algoritmos.pdf

https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=646:documentar-proyectos-java-con-javadoc-comentarios-simbolos-tags-deprecated-param-etc-cu00680b&catid=68&Itemid=188