## Programmer or engineer

| |
|---|
| engineer = analytic mindset, oversee the whole software development cycle |
| programmer = implement and debug |

## Water fall model

| | |
|---|---|
| program specification | project description/requirements |
| program design | plan to solve the problems listed, no code at all, language independent |
| implementation | coding in java |
| testing and debug | testing case(end point, in range, out of range, special value/invalid) |
| | compiler(java checked Exception)/runtime(java unchecked,error)/business logic errors(in corrected results) |
| | robustness:deal bad input ( input correct ->correct answer, invalid input ->no crash/error/exception |
| maintenance | upgrade, new features,document for new comers |

infinite loop is a run time error because it will eventually eat up all memory.
divided by zero for int(runtime exception) and float (infinite, when output is known)
platform dependency and efficiency not belong to robustness.

## object oriented program design

| |
|---|
| identify classes -- noun |
| identify behavior --verb |
| find relation between classes |
| write API for each class ( pubic method header) |
| implement methods |

## identify classes

| | |
|---|---|
| lower level component | Connect4 board, game rule |
| collection of lower level components | players |
| control class | controller ( setup player and board) |
| GUI | Display |

## identify behavior

| | |
|---|---|
| all verbs | |
| match behavior with class carefully and logically | This could be in test |
| encapsulation, only expose what necessary | |
| find suitable data structure( fields, properties) | |

## implementing methods

| | |
|---|---|
| procedural abstraction/stepwise refinment | helper methods (repeated code, smaller method body,easy read) |
| information hiding | instance variable and helper method to be private |
| stub method | for test, input and return correctly |
| algorithm | step by step procedure to solve problem. understand the algorithm before coding |
| | logically correctness,CPU time, memory needed,easier to maintain/understand |

## determine relationships between classes

| | |
|---|---|
| inheritance | is-a |
| composition | has-a |
| UML Model | |
| rectangular | <<abstract>><<interface>> |
| up arrow | is a |
| up arrow + dotted line | interface is a |
| down arrow | has a |

use is-a and has-a to test if class design is ok or not.

## implementing classes

| | |
|---|---|
| bottom up | independent class -->lower lever first -->major components |
| top down | main class ->step wise refinement and add detail classes |

design is usually top down, implementation usu bottom up is easier.

## program analysis

| | |
|---|---|
| assertion | precondition( what's is true before a method,loop, block run - input), post condition (the status after the run - output) |
| efficiency | cpu time, memory |

best case; worst case;average case



Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com

| Vocabulary summary | |
|---|---|
| software development | writing a program |
| object-oriented program | uses interacting object |
| program specification | description of a task |
| program implementation | the code |
| test data | input to test a program |
| program maintenance | keep program working and up to date |
| top down development | implement main class first, subsidiary classes later |
| bottom up development | implement lowest level classes, independent class first |
| driver class | used to test other class, contains main() |
| inheritance relationship | is a relationship between classes |
| composition relationship | has a relationship between classes |
| inheritance hirarchy | inheritance relationship shown in a tree like structure |
| UML diagram | tree like representation of relations between classes |
| data structure | java construt for storing data |
| encapsulation | combining data field and method in a class |
| information hiding | use private to restrict access |
| step wise refinement | breaking methods into smaller methods |
| procedural abstraction | use helper method |
| algorithm | step by step process to solve a problem |
| stub method | dummy method called by another method being tested |
| debugging | fixing errors |
| robust program | screens out bad input |
| compile time error | usu a xyntax error; prevent from compiling |
| syntax error | bad language usage |
| runtime error | occurs during eecution |

| Vocabulary summary (cont) | |
|---|---|
| exception | run time error thrown by java method |
| logic error | program runs but does the wrong things |
| independent class | donot use other classes of the program in its code |



Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com