Computational Thinking, Programming and Algorithms

Keywords

2.1 ALGORITHMS

COMPUTATIONAL THINKING:

ABSTRACTION

Picking out the important bits of information

DECOMPOSITION

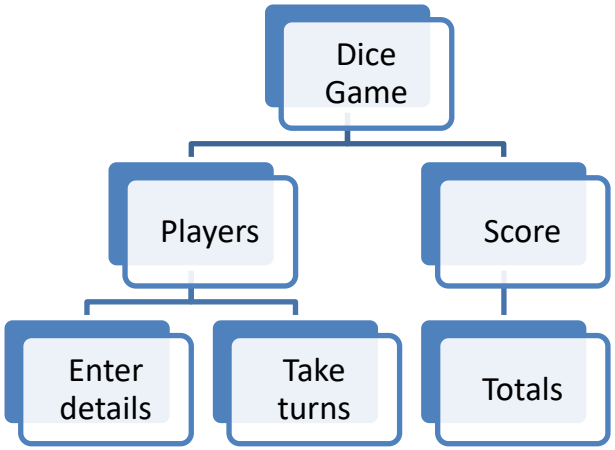
Breaking down problems into smaller problems.

ALGORITHMIC THINKING

Coming up with an algorithm to solve a problem.

IDENTIFY I,P,O,P

STRUCTURED DIAGRAMS



CREATE, INTERPRET, CORRECT, COMPLETE, REFINE ALGORITHMS

IN PSEUDOCODE, FLOWCHARTS, PYTHON

IDENTIFY ERRORS

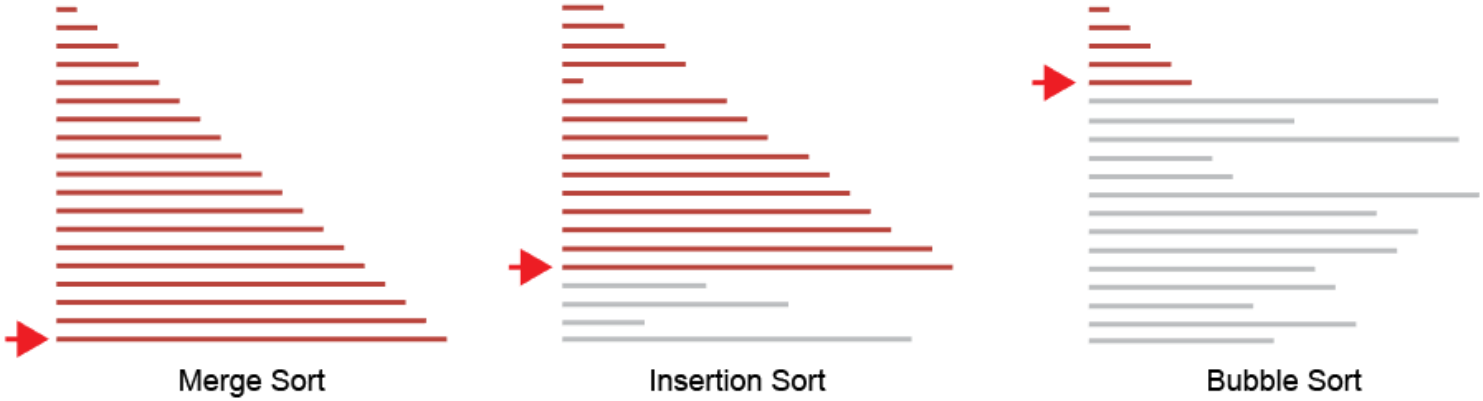
TRACE TABLES

SEARCHING ALGORITHMS:

BINARY SEARCH

LINEAR SEARCH

SORTING ALGORITHMS:



BUBBLE SORT

MERGE SORT

INSERTION SORT

2.2 PROGRAMMING TECHNIQUES

VARIABLES

A value stored in memory that CAN change while the program is running.

CONSTANTS

A value stored in memory that CANNOT change WHILE the program is running.

OPERATORS

>,<,<=,>=,==,!=,<>

INPUTS

OUTPUTS

ASSIGNMENTS

Sets and/or re-sets the value stored in the storage location(s) denoted by a variable name; in other words, it copies a value into the variable.

THREE PROGRAMMING CONSTUCTS:

SEQUENCE

More than one instruction to be followed in order.

SELECTION

if/else

```
if entry=="a" then
   print("You selected A")
elseif entry=="b" then
   print("You selected B")
else
   print("Unrecognised selection")
endif
```

ITERATION (LOOPS)

FOR LOOP (COUNT CONTROLLED LOOP)

WHILE LOOP (CONDITION CONTROLLED LOOP)

ARITHMETIC OPERATIONS

+	Addition eg x=6+5 gives 11
-	Subtraction eg x=6-5 gives 1
*	Multiplication eg x=12*2 gives 24
/	Division eg x=12/2 gives 6
MOD	Modulus eg 12MOD5 gives 2
DIV	Quotient eg 17DIV5 gives 3
^	Exponentiation eg 3^4 gives 81

BOOLEAN OPERATORS

==	Equal to
!=	Not equal to
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to

DATA TYPES: INTEGER, FLOAT(REAL), BOOLEAN, CHARACTER, STRING, CASTING

STRING MANIPULATION

Example:

```
someText="Computer Science"
print(someText.length)
print(someText.substring(3,3))
```

Will display:

```
16
put
```

FILE HANDING OPERATIONS:

OPEN

```
myFile = open("sample.txt")
```

READ

```
myFile = openRead("sample.txt")
```

WRITE

```
myFile = openWrite("sample.txt")
myFile.writeLine("Hello World")
```

CLOSE

```
myFile.close()
```

RECORDS USED TO STORE DATA

A collection of fields.

Allows multiple values of the different data types which are related values. E.g. table.

SQL TO SEARCH FOR DATA

Structured Query Language (SQL) is used for database queries.

SELECT (ColumnName1, ColumnName2, …)

FROM TableName

[WHERE ColumnName2=<condition1> [AND/OR/NOT

…<condition2> OR ColumnName4 LIKE <%pattern%>]

[ORDER BY (ColumnName, …) DESC/ASC];

1D AND 2D ARRAYS

```
array names[5]
names[0]="Ahmad"
names[1]="Ben"
names[2]="Catherine"
names[3]="Dana"
names[4]="Elijah"

print(names[3])
```

Example of 2D array:

```
array board[8,8]
board[0,0]="rook"
```

SUB PROGRAMS: FUNCTIONS AND PROCEDURES

RANDOM NUMBER GENERATION

2.3 PRODUCING ROBUST PROGRAMS

DEFENSIVE DESIGN CONSIDERATIONS:

ANTICIPATING MISUSE

AUTHENTICATION

INPUT VALIDATION

MAINTAINABILITY

Use of sub programs

Naming conventions

Indentation

Commenting

THE PURPOSE OF TESTING

ITERATIVE TESTING

FINAL/TERMINAL TESTING

SYNTAX ERRORS

LOGIC ERRORS

SELECTING TEST DATA

Normal

Boundary

Invalid

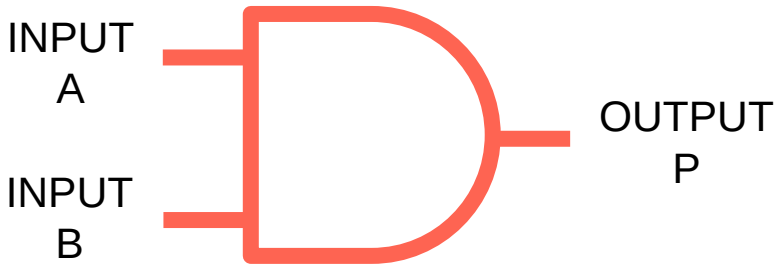
Erroneous

REFINING ALGORITHMS

2.4 COMPUTATIONAL LOGIC

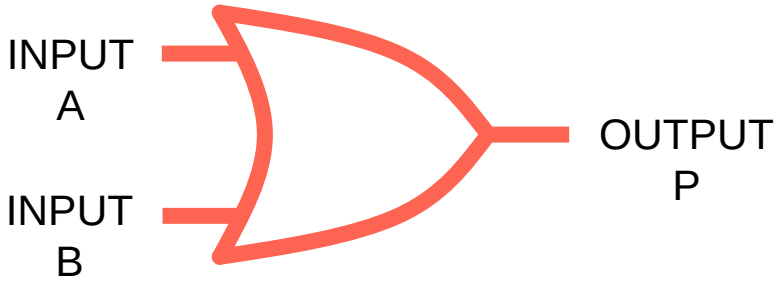
LOGIC DIAGRAMS:

AND



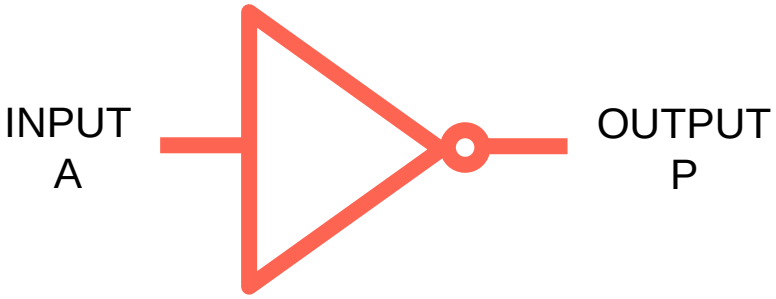
Logic statement: P = A AND B

OR



Logic statement: P = A OR B

NOT



Logic statement: P = NOT A

TRUTH TABLES:

AND

AND (conjunction)		
INPUT		OUTPUT
A	B	$A \wedge B$
T	T	T
T	F	F
F	T	F
F	F	F

OR

OR (disjunction)		
INPUT		OUTPUT
A	B	$A \vee B$
T	T	T
T	F	T
F	T	T
F	F	F

NOT

NOT (negation)	
of ¬ A	
A	$\neg A$
T	F
F	T

BOOLEAN OPERATORS USING: AND, OR AND NOT

2.5 PROGRAMMING LANGUAGES & IDE's

LOW & HIGH LEVEL PROGRAMMING LANGUAGE

TRANSLATORS

Purpose? Examples?

ASSEMBLER, COMPILER, INTERPRETER

Characteristics?

TOOLS IN AN IDE:

EDITORS, ERROR DIAGNOSTICS, RUN-TIME ENVIRONMENT, TRANSLATORS