Unit 2.2 Problem Solving and Programming

Variables

- Variables can be either global or local scope.
- Scope refers to the section of code where the variable can be accessed.
- A local variable in a subroutine has precedence over a global variable with the same name.

Local Variables

- Can only be accessed within the subroutine where they were defined.
- Multiple variables with the same name can exist in different subroutines.
- Are deleted when the subroutine ends.
- Ensures subroutines are self contained.

Global Variables

- Can be accessed through the whole program.
- Used for values needed throughout the program.
- Risk the variable is unintentionally edited.
- Uses memory for longer.

Problem Recognition

- Stakeholders say what they need from the solution.
- This information is used to produce a clear list of system requirements and a definition of the problem.
- We may consider the strengths and weaknesses of a current system.
- We may consider the required inputs, outputs and the volume of stored data.

Modularity

- Large or complex programs can be split into smaller self contained modules.
- This makes it easier to divide tasks between a team and manage the project.
- It simplifies maintenance since each component can be handled individually.
- It improves the reusability of code.
- Top Down (Stepwise) Refinement
- A technique used to modularise programs.
- The problem is broken into sub problems until each sub problem is a single task.
- Modules form blocks of code called subroutines.

Programming Constructs

- Sequence – Code is executed line by line from the top down.
- Breaching – A block of code is run only if a condition is met using IF and ELSE statements
- Count Controlled Iteration – A block of code is run a certain number of times. Uses FOR, WHILE or REPEAT UNTIL statements.
- Condition Controlled Iteration – A block of code is run while or until a condition is met. Uses FOR, WHILE or REPEAT UNTIL statements.

Integrated Development Environment

- Programs used to write code.
- Contains a set of tools which make it easier for programmers to write, develop and debug code.
- May include stepping, variable watching, breakpoints, source code editor and debugging tools.

Recursion

- When a subroutine calls itself during execution.
- Continues until a stopping condition is met.

Advantages

- Requires fewer lines of code.
- Easier to express functions such as factorials recursively.

Disadvantages

- Risk of stack overflow if memory runs out.
- Often challenging to trace and locate errors.

Object Orientated Techniques

- Object oriented programming languages use classes.
- A class is a template for an object.
- An object is an instance of a class.
- It defines the behaviour and state of objects.
- Object state uses attributes.
- Object behaviour uses methods.
- Encapsulation is used to edit attributes.
- Top down design applies encapsulation to modules.
- Modules are built to be self contained and reusable.

Can a Problem be Solved by Computational Methods?

- Not all problems can be solved in this way.
- Some may need too many resources or time.
- Problems which can be solved using algorithms lend themselves well to being solved via computational methods.
- We must identify whether the problem can be solved using computational methods before we attempt to solve it.

Divide and Conquer

- A problem solving technique with three parts.
- Divide - halve the size of the problem with each iteration.
- Conquer - solve the subproblems.
- Merge - combine the solutions.
- It is applied in binary search, quick sort and merge sort.
- It is a quick way to simplify complex problems.

Problem Decomposition

- The problem is broken down into smaller subproblems.
- This is repeated until each subproblem can be represented using a single subroutine.
- This reduces the complexity of the problem and makes it easier to solve.
- It enables programmers to see which areas can be solved using pre-existing libraries or modules.
- It makes the project easier to manage.
- Subproblems can be assigned to different specialist teams or individuals.
- Modules can be designed and tested individually before being combined.
- It makes it possible to develop modules in parallel and therefore finish more quickly.
- It is easier to debug the code and locate errors.

Abstraction

- Represents real world entities using computational elements.
- Excessive details are removed to simplify the problem.
- This may then match a problem which has previously been solved.
- Existing modules, functions or libraries can then be used to solve the problem.
- Levels of abstraction divide a complex problem into smaller parts.
- Different levels can be assigned to teams whilst hiding details of other layers.
- This makes the project easier to manage.
- Abstraction by generalisation groups together sections with similar functionality.
- This allows segments to be coded together, saving time.

Problem Solving Strategies

Backtracking

- Uses algorithms, often recursively.
- Builds a solution methodically.
- Based on paths which have been visited and found to be correct.
- The algorithm backtracks to the previous stage if an invalid path is found.

Data Mining

- Identifies patterns or outliers in large data sets, often collected from multiple sources.
- These data sets are known as big data.
- It spots correlations between data and other trends which might not be easy to see.
- Can be used to make predictions about the future.
- A useful tool to assist in business and marketing.

Heuristics

- A non optimal or rule of thumb approach.
- Used to find an approximate solution to a problem.
- Used where the standard solution takes too long.
- Does not produce a 100% accurate or complete solution.
- Provides an estimate for intractable problems.
- Performance Modelling
- Mathematical method to test loads on systems.
- A cheaper and less time consuming method of testing applications.
- Used for safety critical systems where a trial run can't be carried out.

Pipelining

- Modules are divided into individual tasks.
- Tasks are developed in parallel.
- Allows faster completion.
- The output of one process is often the input of another.
- Often used in RISC processors, which perform different parts of the Fetch-Decode-Execute cycle at the same time.

Visualisation

- Presenting data using charts or graphs.
- Makes it easier for humans to understand.
- Allows trends or patterns to be more easily identified.

Functions and Procedures

- Named code blocks which perform a particular task.
- Functions must always return a single value.
- Procedures do not have to return a value.
- Parameters can be passed to them by either reference or value.

Passing by Reference

- The address of the parameter only is given to the subroutine.
- The subroutine works on the value at the given address.

Passing by Value

- A copy of the value is passed to the subroutine.
- The original value is unchanged.
- The copy is deleted at the end of the subroutine.
- Exam questions will use this technique unless told otherwise.
- Exam questions will use the format function function(x:value, y:value)