OCR Computer Science A Level

2.1.3 Thinking Procedurally

Concise Notes

Specification:

2.1.3 a)

- **Identify the components of a problem**

2.1.3 b)

- **Identify the components of a solution to a problem**

2.1.3 c)

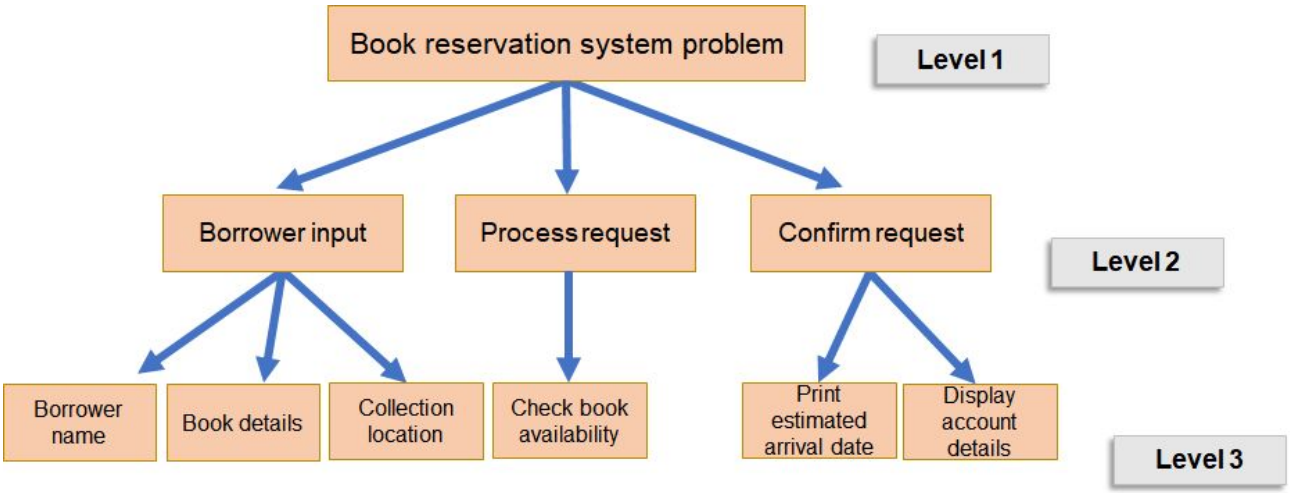
- **Determine the order of the steps needed to solve a problem**

2.1.3 d)

- **Identify sub-procedures necessary to solve a problem**

Identify the components of a problem

- Breaking a problem down is the first stage of thinking procedurally.
- This process is called problem decomposition.
- This breaks a large, complex problem down into smaller subproblems which can be solved more easily.
- The project becomes easier to manage and can be divided between a team
- Top-down design, also known as stepwise refinement, is commonly used to do this



- This divides problems into levels of complexity.
- Problems are broken down into subproblems until each subproblem is a single task
- Each subproblem can then be solved using a single subroutine.
- Subroutines can be developed and tested separately, so they are self-contained.

Components of a solution and Sub-procedures

- Details about how each component is implemented are considered.
- Just as we broke down the problem, we now build-up to its solution.
- Need to consider the lowest-level components from top-down design and how they can best be solved.
    - Can this be implemented as a function or a procedure?
    - What inputs are required?
    - What output does the subroutine need to produce?
- Tasks which can be solved using an already existing module are identified

Order of steps needed to solve a problem

- When constructing the final solution, thinking about the order in which operations are performed is important.
- Programs may require certain inputs to be entered in a particular order by the user before processing can occur.
- Inputs need to be validated, and this must occur before this data is used.
- In some cases, it may be possible for several subroutines to be executed simultaneously depending on the data and inputs the subroutine requires.
- Programmers should decide on the order in which subroutines are executed, and how they interact with each other, based on their role in solving the problem.
- Programs should also be built so operations cannot be carried out in an order that will raise an error or does not make logical sense.