Elements of Good Programming Design

Naming Variables and Constants

- Give variables meaningful names
- Follow naming conventions
- Variables must be one word – no spaces
- Variables must start with a letter
- Use a different convention for constants – for example all capital letters.

More on Modularization

- Main program can also be referred to as mainline logic
- Generally consists of:
  - Housekeeping Tasks that perform task at the beginning including variable and constant declarations, displaying instructions to the user, displaying report headings, opening files, and inputting the first data item (priming input).
  - Detail Loop Tasks that perform core processing
  - End of Job Task including closing files, displaying final totals and messages and other cleanup tasks.

More on Modularization

- Modules are said to **encapsulate** the details of a particular process.
- Modules should contain the logic needed to perform a single function – a concept known as **functional cohesion**.
- A module that determines and deducts insurance premiums only is cohesive. One that also computes taxes would be less cohesive. Cohesive functions are more easily reusable.

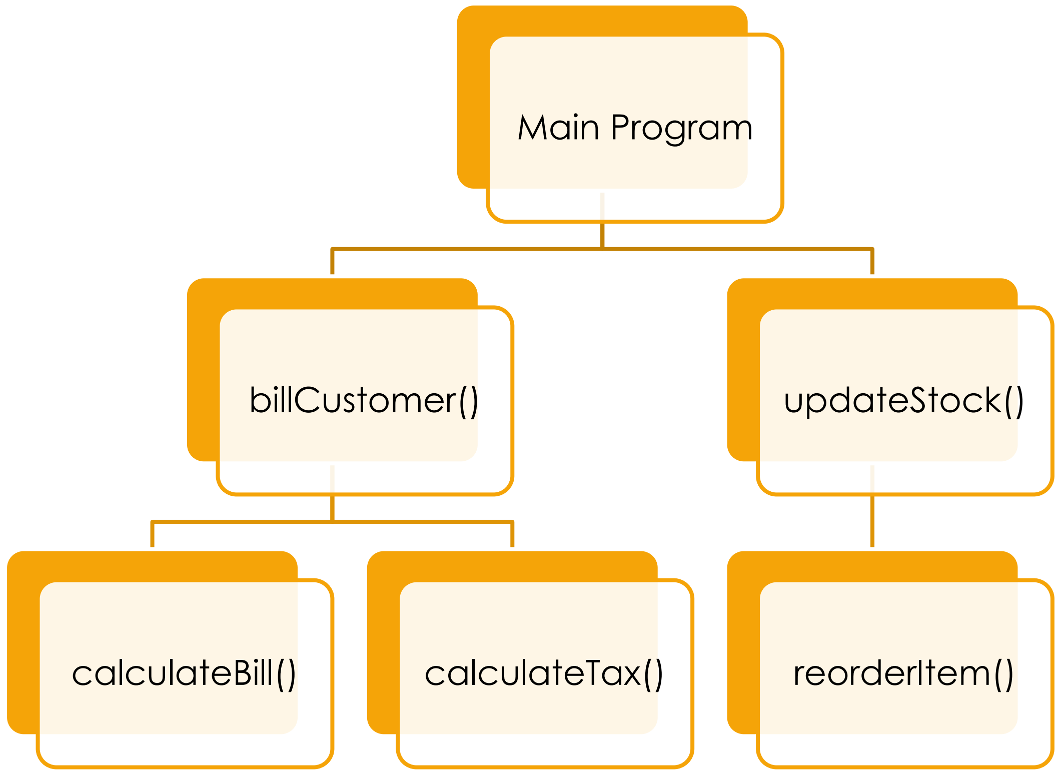
Variables and Scope

- Where a variable is declared determines its **visibility** or **scope**, that is where it can be used by the program.
- A variable declared in a module is **local** to the module. The variable comes into existence when control passes to the module and ceases to exist when the module returns control to the main program.
- This makes the module more **portable** – it does not have a dependence on the main program and can be used elsewhere.
- **Global** variables can be declared at the program level, and thus are visible to the entire program including the main program and any modules that are defined within it.

Hierarchy Charts

- **Hierarchy charts** are another tool that programmers use to show how modules are related to one another.
- It identifies **which** modules are part of the program and **which modules call other modules**.
- They do not tell you what each module does, why they are called or in what order they are called. You must use pseudocode or flow charts for that information.
- Hierarchy charts are helpful in understanding the organization of your program and resemble the org chart of a company.

Hierarchy Chart



Features of Good Programs

- Use program comments where appropriate
- Choose meaningful and easily understandable identifiers (variable and constant names, module names, etc.)
- Design clear statements
- Write clear prompts to the user and echo the user’s input
- Develop and maintain good programming practices

Features of Good Programs

- Program comments serve as internal documentation
- Explain complicated calculations, assumptions, programming decisions, what variables are used for, etc.
- Necessary when working with other team members and useful when modifying code that you have not worked on in awhile.

Features of Good Programs

- For a long statement, use line breaks at natural places to make code more readable.
- Do not put more than one statement on a line.
- Use prompts to users that clearly indicate the expected and valid input.
- Echo the input after the user enters it in subsequent prompts or when showing results.
  - Example – "The sum of 12 and 14 is 26" rather than just "The sum is 26"