Understand the Programming Process

Objective/Essential Standard

- Essential Standard: 2.00 Understand the Solution Development Process

- Indicator: 2.01 Understand the Programming Process. (3%)
The Programming Process

- A computer program is a list of instructions that contain data for a computer to follow. Different programs are written with different languages.

- A five step programming process is outlined in this section.

1. Identify the Problem
2. Design the Solution
3. Write the Program
4. Test the Program
5. Document and Maintain the Program

1. Identify the Problem

- Two parts to this step

  1. Requirements
     - What is needed to be part of the solution
     - Define what your program needs to do (the desired output)
     - Understand the reason for creating the program and who will be using it.
     - Examine the data to be processed to ensure the program will handle data requirements.

  2. Specifications
     - What does your program need to do to fulfill requirements.
     - Determine the needs of the users
2. Design the Solution

- The most frequently used design in the programming process is called **Top-Down Design**.

- A solution method is broken down into smaller sub-problems, which in turn are broken down into smaller sub-problems, continuing until each sub problem can be solved in a few steps. This is called **modularization**.
  - In large projects, sub-problems may be assigned to different programmers, or teams of programmers, who are given precise guidelines about how their sub-problems fit into the overall solution design.
  - These individuals or teams are then given the opportunity to design the solution to their sub-problem as they deem best.

2. Design the Solution

- One method of designing a solution to a problem or sub-problem is to create an algorithm.

- An **algorithm** is a set of steps that create an ordered approach to a problem solution.

- Several methods for designing a programming algorithm exist.
  1. An algorithm can be written in plain **English or outline** form.
  2. An method called **pseudocode** may be used to create an algorithm.
  3. A third method of creating an algorithm is called **flowcharting**.
3. Write the Program

- Coding the program should include:

  1. Choosing the correct programming language that will best suit the needs and desired outcome of the users.
     - Different languages are designed to handle data in different ways.
     - Some languages are designed to handle numeric processing, others are better suited to handle textual data.
  2. Once the programming language has been determined, follow the syntax of the programming language precisely.
  3. Follow good programming style.
     - For example, in Visual Basic, label names should begin with lbl.
     - Agreed upon “rules” to make reading code easier.

4. Test the Program

- After coding the program, testing should occur in several steps:

  1. First, the program should be tested for all syntax errors (debugging – running your program).
  2. When the program executes or compiles correctly, it should be checked for logic errors. Just because it runs does not mean it is working correctly. Run some diagnostic tests with sample data to be certain that the data which is input is handled correctly and produces the desired output.
  3. Beta test your program by using some real world data.
5. Documentation and Maintenance

- Documentation is crucial for user information as well as for programmer understanding.

- Use documentation to create **User Guides** so users will be certain they are adhering to the design of the program.
  - For example, an input field asking for your birthday might also have the desired format included in parentheses next to the input field (mm/dd/yyyy).

- Always apply documentation through comments to your code so you can remember what you are doing and so other programmers can follow the logic of the code.
  - This is also a good practice to follow each time you update or change code written by someone else.

The Programming Process

- Remember programming is a dynamic process.

- Good programmers follow the process.