First steps: Building and Running

Before you can get started learning to program you need to get started using the terminal, get access to the tools you need to build programs, and practice building and running some programs from supplied source code.

Learning Goals

To complete this task, you need to demonstrate that you can do the following:

- **Digital realities**: Describe how programs are like digital realities, and how this relates to abstraction.
- **Computer use**: Access and interact with your computer via a terminal, focusing on the use of commands to:
  - Move between and explore folders (cd, ls, pwd)
  - Copy and move files (cp, mv)
  - Build and run programs (dotnet new console, dotnet build, dotnet run)
- **Building programs**: Create, run, and debug programs using the dotnet tools, the terminal, and Visual Studio Code.

Focus

As you work through this task, focus on the following aspects of the unit:

- **Programming process:** Focus on building your confidence in using the programming tools to build, run, and debug code. You will use this over and over as you start building your own programs.
- **Coding:** You can think of shell command as coding in the terminal. Focus on the general structure of these and memorise the main commands/tools that you can use to work with files and build and run programs.

- **Professional Characteristics:** Attention to detail is one of the main characteristics underpinning success in computing. Computers are unintelligent, so even things like different letter cases can result in commands not working. Focus on paying attention to the detail for each command and aspect as you work through the material.

Your Task

For this task you will need to submit the following:

- A PDF document containing:
    - Summaries and reflections with your descriptions of digital realities and the role of abstraction in programming.
    - At least three (3) Screenshots:
        - One of you using the terminal,
        - One of you running Hello World in Visual Studio Code, and
        - One of you using the debugger to step through a program.
    - Learning Journey and Resources

1. Complete Learning Activities

Work through these steps to develop and demonstrate your understanding. Aim to demonstrate, to yourself and others, that you have achieved the learning goals.

1. Everything you need to gain an overview of programming is in Chapters 0 Introduction and Chapter 1 Digital Realities from Part 0 of the Programmer's Field Guide. Chapter 2 Computer Use and Chapter 3 Building Programs provide details on the use of shell commands (in the terminal) to work with files and navigate between folders.

Tip

Read these chapters *actively*. This is not like reading a novel. Have your computer with you and test out the ideas as you read through the chapters. You may need to go back and forth between sections until you feel you understand how the related concepts and tools can be used.

2. Demonstrate the use of the terminal to interact with files and folders. Show how you setup your development environment and installed the tools that you needed to start building and running programs.

Tip

There are many shell commands in these chapters, but to get started you only need a few. Start with the commands to move between folders and build and run your programs. Perhaps a cheat-sheet to help you remember some of the main points. You can include this in your submission as evidence of your learning.

3. Follow the steps from the Programmer's Field Guide to create and run the Hello World program. **Capture** a screenshot of the program running in the terminal in Visual Studio Code for your submission.
4. Use the Terminal to create projects and run at least two of the following sample programs. In each case, copy the code from the supplied link and paste it into your *Program.cs* file within a dotnet project. You should then be able to build and run the program. Many of these use SplashKit, so make sure your project is setup to include that library.
   - Graphical Hello World
   - Hello World with Images and Sounds
   - Airspeed Velocity
   - Water Bottle Visualiser
   - Change Calculator (version 2)
   - Fly Escape

5. Open the code for at least one of these programs in Visual Studio Code, and step through it with the debugger. Test out breakpoints, and step through the code line by line as the computer runs it. **Capture** some screenshots of this in action for your submission.

Tip: Review the Task Format guidance

The unit site contains a document on the general structure of these module-based tasks. This document provides additional guidance on how to approach creating your summaries and learning journey. Make sure to review that to help guide you in preparing your submission.

6. Prepare your summary, making sure to cover all learning goals and related concepts. Remember that this is a personal summary that demonstrates your understanding of the concepts.

Caution: Acceptable use of Artificial Intelligence

It is acceptable to use AI to improve your style, as you would be using a spell checker to improve your spelling. Do **not** ask an AI to write the summary for you: this is cheating because it does not demonstrate that you understand anything.

7. Prepare your reflections by responding to the following:
    - How do you know you have achieved the learning goals?
    - What is the most important thing you learned from this and why?
8. Capture your learning journey and collate your evidence of study and practice.

2. Upload Your Submission & Engage with Feedback

Mark the task as **Ready for Feedback** and upload the required files. Make sure to keep copies of these in case you need to resubmit. Then engage with the feedback you receive and get the task Complete.

Tip: Initiate feedback

When you submit your work, you can add a comment to your tutor. It is a great idea to use this to ask for specific feedback on your work. This can help make sure you receive the most effective feedback to support your learning.

How to Succeed in SIT102

Attend classes: The classes give you access to tutors and other students working on the same topics as you. They provide a great environment to engage with the unit, clarify difficulties and accelerate your progress. Students who attend class regularly are much more likely to find the OnTrack tasks easier to complete, stay up to date, and generally find the unit quite manageable. Students who miss classes tend to take much longer to complete OnTrack tasks, find them more difficult to complete, and find themselves to be less knowledgeable about key topics.

Stay up to date with your tasks: we have OnTrack tasks due every week, so it helps to complete tasks in a timely fashion so you don't have a large collection of tasks to complete all at once (which can happen quite quickly) and generally leads to poor quality work and a fail result in the unit.

Engage and be involved: successful students are those that ask their tutors questions, are active on the MS Team channel or Discussion forum, and reach out for help when they need. You'll learn a lot from these interactions, so make yourself known to your tutors and classmates.