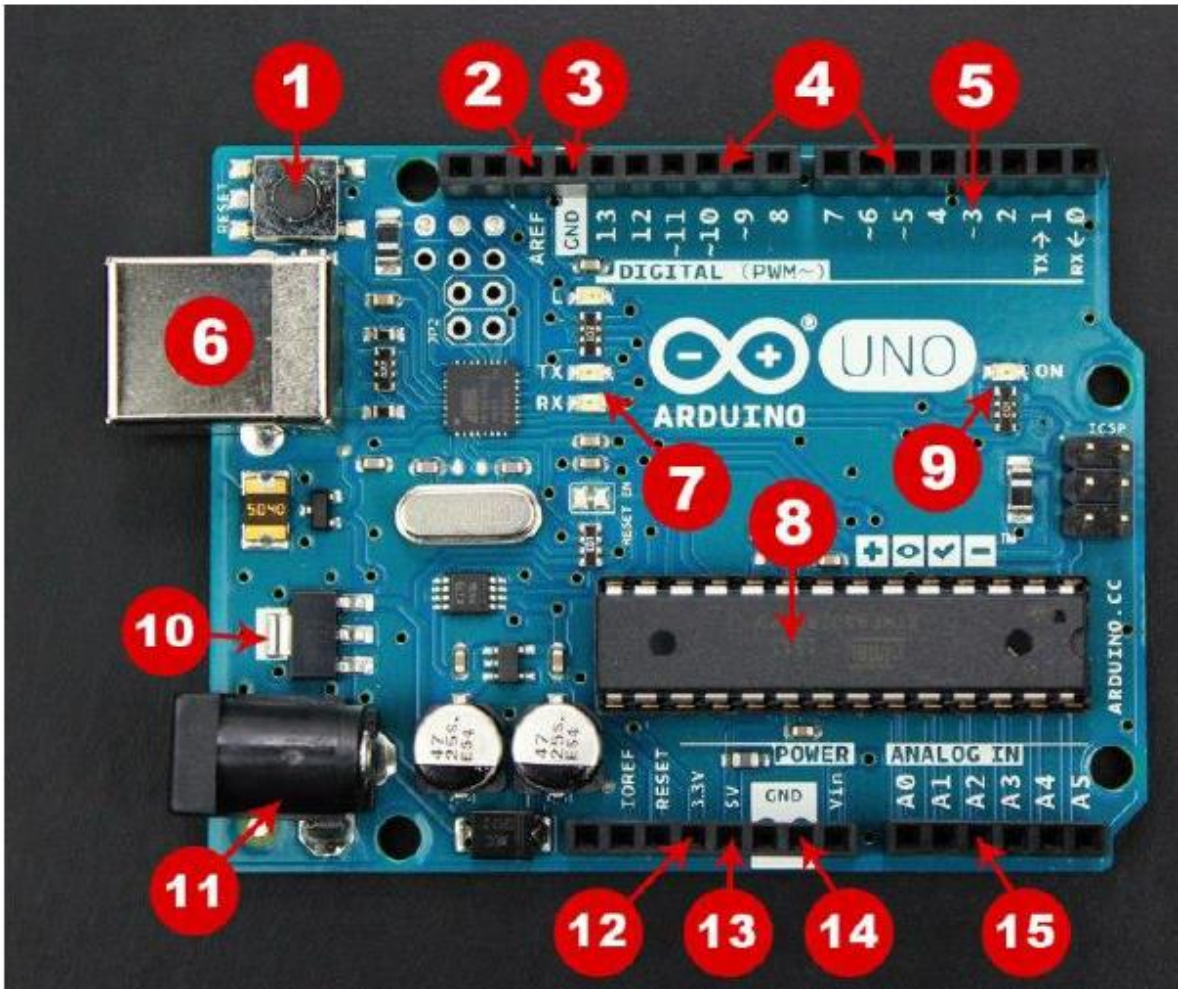
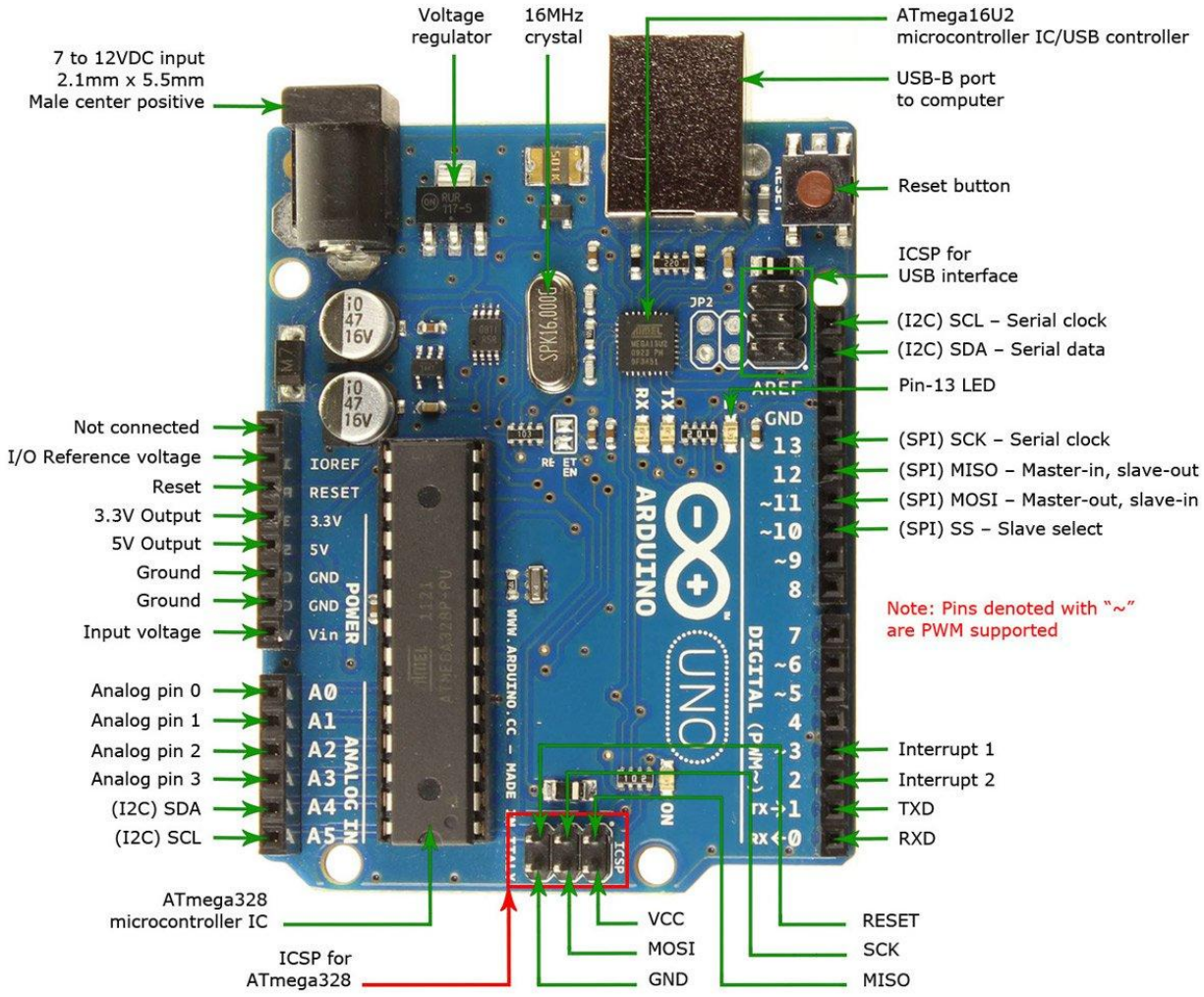
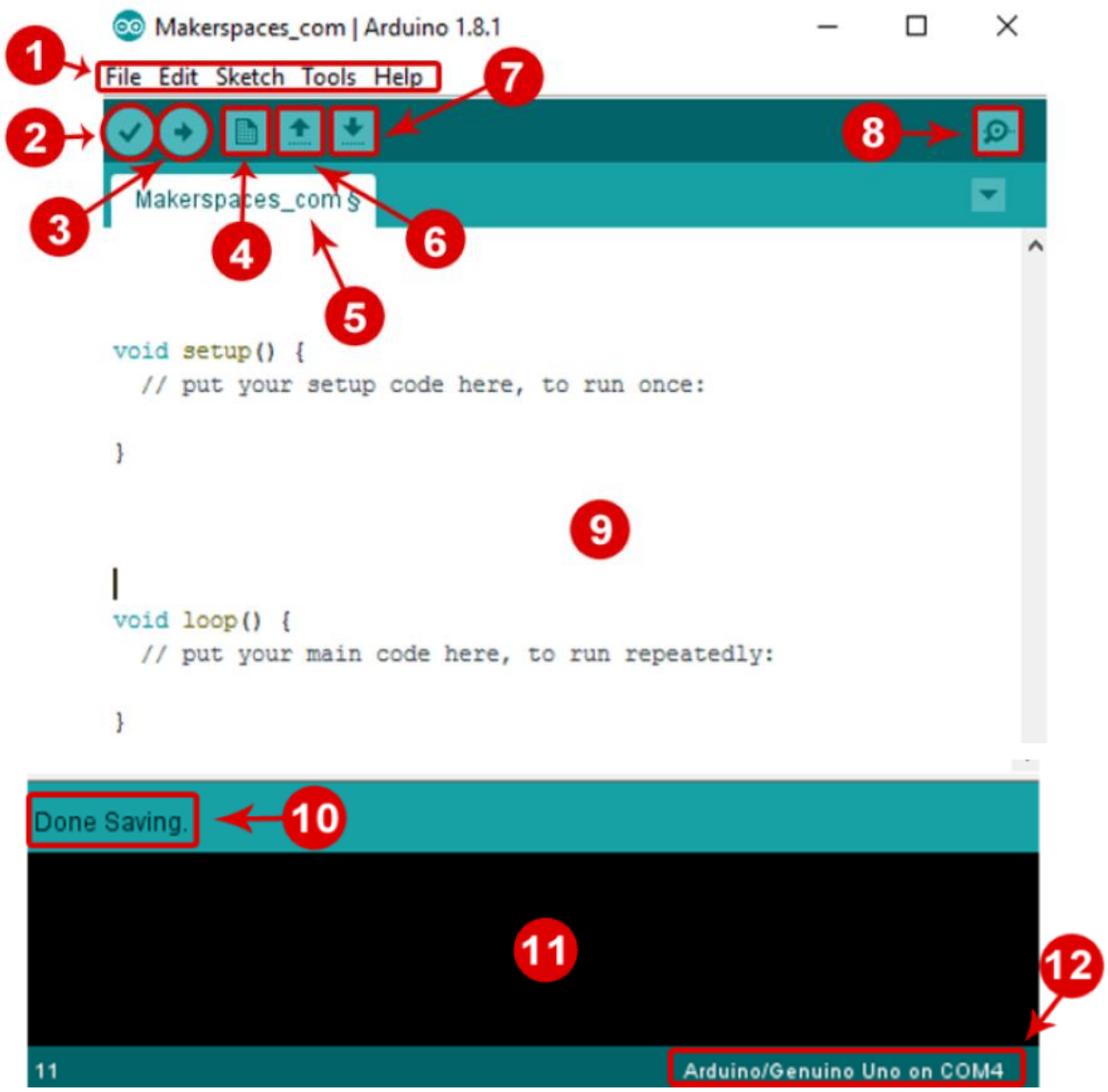


1. **Reset Button** – This will restart any code that is loaded to the Arduino board
2. **AREF** – Stands for "Analog Reference" and is used to set an external reference voltage
3. **Ground Pin** – There are a few ground pins on the Arduino and they all work the same
4. **Digital Input/Output** – Pins 0-13 can be used for digital input or output
5. **PWM** – The pins marked with the (~) symbol can simulate analog output
6. **USB Connection** – Used for powering up your Arduino and uploading sketches
7. **TX/RX** – Transmit and receive data indication LEDs
8. **ATmega Microcontroller** – This is the brains and is where the programs are stored
9. **Power LED Indicator** – This LED lights up anytime the board is plugged in a power source
10. **Voltage Regulator** – This controls the amount of voltage going into the Arduino board

11. **DC Power Barrel Jack** – This is used for powering your Arduino with a power supply
12. **3.3V Pin** – This pin supplies 3.3 volts of power to your projects
13. **5V Pin** – This pin supplies 5 volts of power to your projects
14. **Ground Pins** – There are a few ground pins on the Arduino and they all work the same
15. **Analog Pins** – These pins can read the signal from an analog sensor and convert it to digital

Illustrate the pin diagram of the Arduino





1. **Menu Bar:** Gives you access to the tools needed for creating and saving Arduino sketches.

2. **Verify Button:** Compiles your code and checks for errors in spelling or syntax.

3. **Upload Button:** Sends the code to the board that's connected such as Arduino Uno in this case. Lights on the board will blink rapidly when uploading.

4. **New Sketch:** Opens up a new window containing a blank sketch.

5. **Sketch Name:** When the sketch is saved, the name of the sketch is displayed here.

6. **Open Existing Sketch:** Allows you to open a saved sketch or one from the stored examples.

7. **Save Sketch:** This saves the sketch you currently have open.

8. **Serial Monitor:** When the board is connected, this will display the serial information of your Arduino

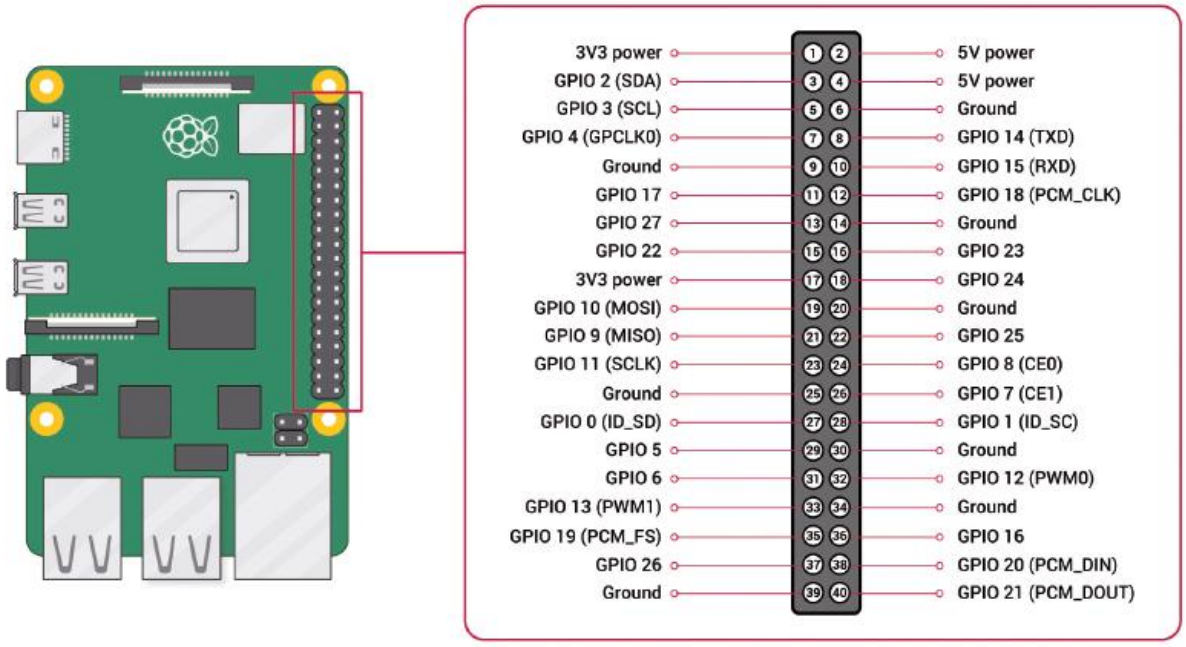
9. **Code Area:** This area is where you compose the code of the sketch that tells the board what to do.

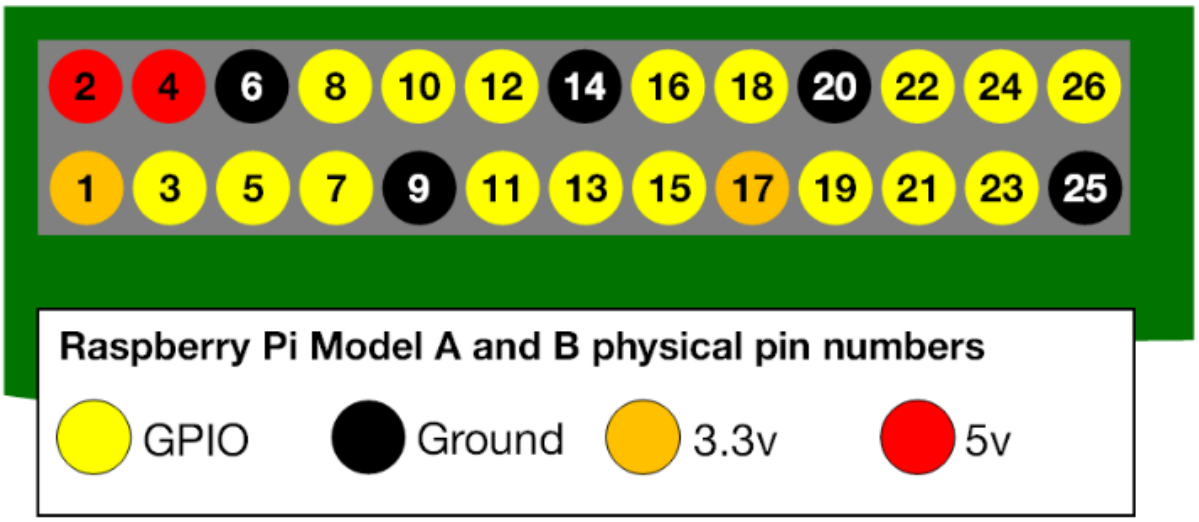
10. **Message Area:** This area tells you the status on saving, code compiling, errors and more.

11. **Text Console:** Shows the details of an error messages, size of the program that was compiled and additional info.

12. **Board and Serial Port:** Tells you what board is being used and what serial port it's connected to.

Draw Pin Diagram of Raspberry Pi:





Outline the steps involved in installing Thonny and explain how to open and execute a program using it?

Method 1: Using apt (Recommended for Raspbian and Raspberry Pi OS)

1. **Update Package List:** Open a terminal window on your Raspberry Pi or connect via SSH, then update the package list to ensure you have the latest versions of software.

   sudo apt update

2. **Install Thonny:** Once the package list is updated, you can install Thonny using apt.

   sudo apt install thonny

3. **Run Thonny:** After installation is complete, you can start Thonny from the applications menu (Programming > Thonny) or by typing thonny in the terminal.

Method 2: Using Python Package Manager (pip)

If you prefer to install Thonny via Python's package manager pip, you can do so as well. This method may give you a newer version of Thonny than what is available in the Raspbian repositories, but it requires Python and pip to be installed.

1. **Install Dependencies:** If you haven't installed pip, install it first:

   sudo apt install python3-pip

2. **Install Thonny using pip:** Use pip to install Thonny:

   sudo pip3 install thonny

   This command installs Thonny globally on your system.

3. **Run Thonny:** Once installed, you can start Thonny by typing thonny in the terminal.

Method 3: Installing from Thonny Website (GUI Method)

If you prefer a graphical method, you can visit the Thonny website and download the .deb package suitable for Raspberry Pi (usually ARM architecture).

1. **Download Thonny:** Go to the Thonny IDE website (https://thonny.org/) and navigate to the downloads section.
2. **Install the .deb package:** Once downloaded, double-click the .deb package file and follow the graphical installer prompts.
3. **Run Thonny:** After installation, Thonny should appear in the applications menu (Programming > Thonny). You can also start it by typing thonny in the terminal.

Notes:

- **Updating Thonny:** To update Thonny, you can use either apt or pip, depending on how you installed it. For example, sudo apt update followed by sudo apt upgrade thonny will update Thonny if installed via apt.
- **Permissions:** Depending on your setup, you may need to prefix commands with sudo (superuser) to install or update software.

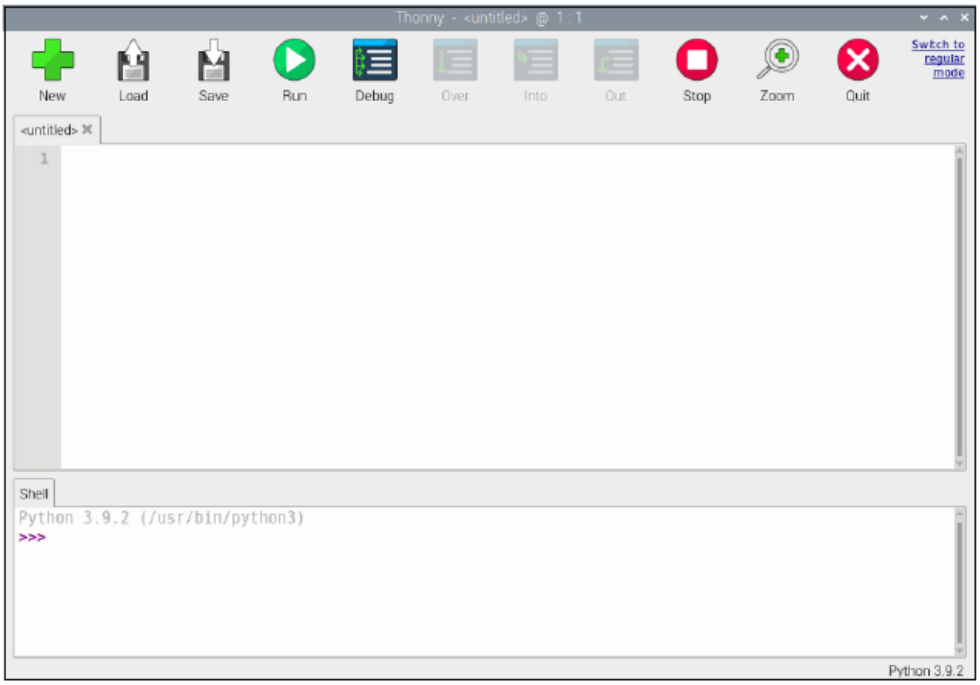


FIGURE 1-7: The Thonny Python IDE.

To actually create a Python program that displays the message Hello, World!, follow these steps:

1. **Press the New button (shown in the margin).** 

Alternatively, you can choose File ➪ New.

Note that you can skip this step if a new window labeled <untitled> is already visible in Thonny.

2. **Type the text** print("Hello, World!")**.**

3. **Click the Save button (shown in the margin)** 

Alternatively, choose File ➪ Save As.

Either way, a Save As dialog box appears. By default, the Save As dialog box opens in the home directory for the current user (/home/pi).

4. I suggest you create a new folder named Python.

To do that, click the New Folder icon in the Save As dialog box (shown in the margin). Then type **Python** and click the Create button.

5. Type hello **as the filename, and then click OK.**

The file is saved using the name hello. Figure 1-9 shows how the hello.py program appears after it has been saved.

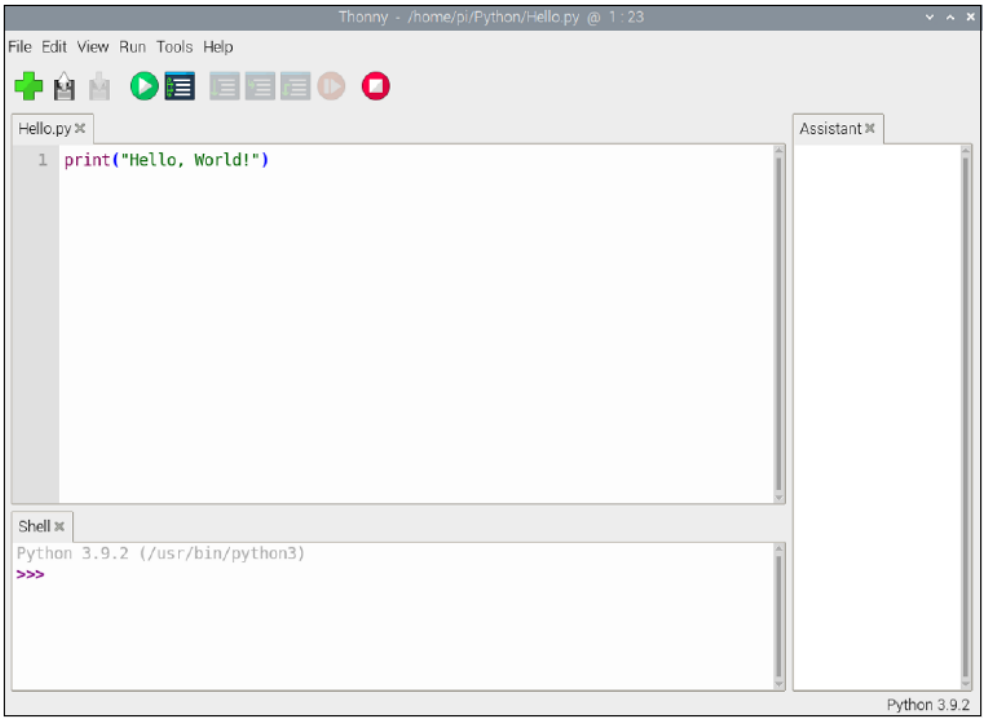


FIGURE 1-9: The hello.py program.

6. Choose Run ➪ Run Module (or press F5).

This action runs the hello.py program. The output from the program (that is, the message Hello, World!) appears in the Python Shell window, as shown

in Figure 1-10.

7. You're done!

Congratulate yourself! You have successfully written your first Python program.

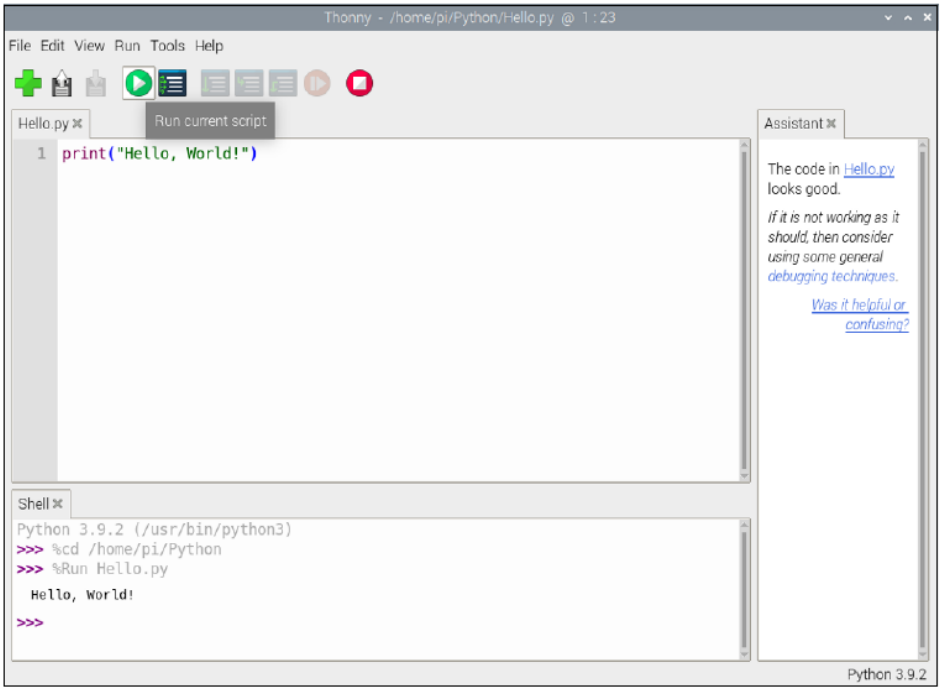


FIGURE 1-10: The Hello, World! Program in action.

To shutdown – open cmd and type poweroff