Nick Greco
Oct 21, 2024
Foundations of Programming: Python
Assignment03

Creating a Simple Student Enrollment System Using PyCharm IDE

Introduction

In this assignment, I will explain the steps involved in creating a script for a basic student enrollment system using PyCharm IDE (Integrated Development Environment). The steps to install and configure this software are briefly outlined. The script expands upon previous assignments by adding logical statements to evaluate conditions and loops to execute segments of code repeatedly. In Python, logical statements and loops are essential for decision-making and controlling the flow of execution in programs. They allow developers to create conditions that determine how code behaves, making it possible to implement complex logic and functionality.

PyCharm Setup

PyCharm is an IDE specifically designed for Python programming. Developed by JetBrains, it provides a comprehensive set of tools and features that streamline the development process for Python applications.

Installation

The following steps can be taken to install PyCharm on a Windows-based computer:

1. **Download PyCharm:**
    a. Go to the JetBrains PyCharm website.
    b. Choose between the Community (free) and Professional (paid) editions and click the corresponding “Download” button.
2. **Run the Installer:**
    a. Once the download is complete, locate the installer and double-click it to run.
    b. If prompted by Windows, confirm that you want to run the installer.
3. **Install PyCharm:**
    a. Follow the installation wizard:
        i. Accept the license agreement.

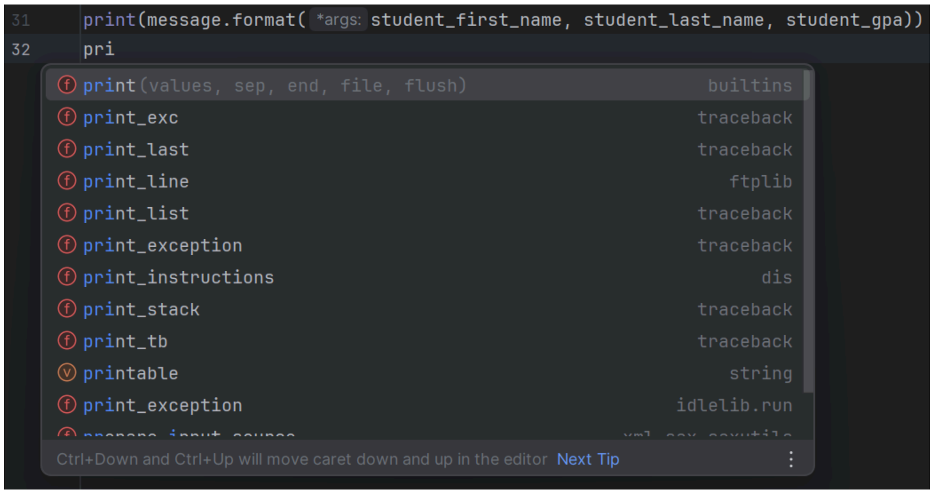
        ii. Choose the installation directory.
        iii. Select additional installation options like creating a desktop shortcut or adding PyCharm to the PATH.
    b. Click Install to begin the installation process.

4. **Complete Installation:**
    a. Once the installation is complete, you can launch PyCharm immediately or from the Start Menu.
5. **Initial Setup:**
    a. On first launch, you might be prompted to import settings from a previous installation.
    b. Choose your theme and complete any additional setup steps as directed.

Productivity Enhancements

Intelligent Code Completion

PyCharm provides intelligent code completion that suggests possible options as you type, including variable names, functions, classes, and methods. It understands context and can suggest completions based on the code you're currently writing. This feature speeds up coding by reducing typing effort and helps prevent errors by ensuring you use correct identifiers and functions. Figure 1 shows an example of intelligent code completion.



```
print(message.format( *args: student_first_name, student_last_name, student_gpa))
pri
```

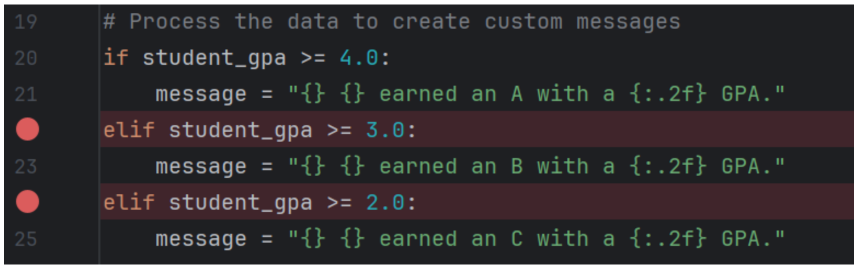
```
print(values, sep, end, file, flush)        builtins
print_exc                                   traceback
print_last                                  traceback
print_line                                  ftplib
print_list                                  traceback
print_exception                             traceback
print_instructions                          dis
print_stack                                 traceback
print_tb                                    traceback
printable                                   string
print_exception                             idlelib.run
```

Ctrl+Down and Ctrl+Up will move caret down and up in the editor
Next Tip

Figure 1: Intelligent Code Completion in PyCharm

Powerful Debugging Tools

PyCharm includes a robust debugger that allows you to set breakpoints, step through code, inspect variables, and evaluate expressions during runtime. These tools help in identifying and fixing bugs efficiently, making debugging less tedious and more intuitive. Figure 2 shows an example of using breakpoints.



```python
# Process the data to create custom messages
if student_gpa >= 4.0:
    message = "{} {} earned an A with a {:.2f} GPA."
elif student_gpa >= 3.0:
    message = "{} {} earned an B with a {:.2f} GPA."
elif student_gpa >= 2.0:
    message = "{} {} earned an C with a {:.2f} GPA."
```

Figure 2: Breakpoints used for Debugging in PyCharm

Virtual Environment Support

PyCharm simplifies the management of Python environments by allowing you to create, manage, and switch between virtual environments directly within the IDE. You can easily configure interpreters for different projects. This feature helps in isolating dependencies for various projects, avoiding version conflicts, and making it easier to work on multiple projects with different requirements.

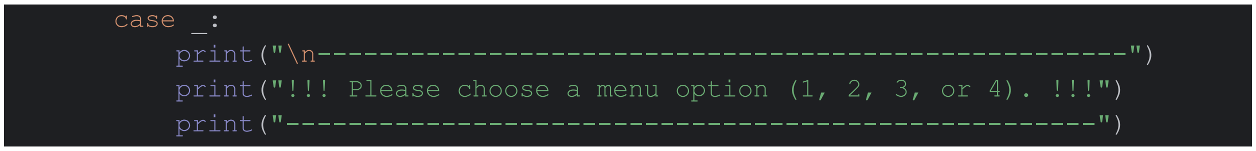
Creating the Script

Overview

Figure 3 below shows the completed script for this assignment.

```python
# --------------------------------------------------------------------------- #
# Title: Assignment03
# Desc:  This assignment demonstrates using conditional logic and looping.
# Change Log: (Who, What, When)
#   N.Greco, 10/19/2024, Created Script
#   N.Greco, 10/21/2024, Changed CSV open from "write" to "append"
# --------------------------------------------------------------------------- #

# Define the Data Constants
MENU: str = """
------ Course Registration Program ------
 Select from the following menu:
  1. Register a Student for a Course
  2. Show Current Data
  3. Save Data to a File
  4. Exit the Program
-----------------------------------------
"""
FILE_NAME: str = "Enrollments.csv"

# Define the Data Variables
course_name: str = ""
```

```
csv_data: str = ""
menu_choice: int = 0
student_first_name: str = ""
student_last_name: str = ""
file_obj: object = None

# Present and Process the Data
while True:
    # Present the Menu
    print(MENU)
    menu_choice = int(input("Menu Selection: "))
    match menu_choice:
        # Input User Data
        case 1:
            student_first_name = input("\nStudent's First Name: ")
            student_last_name = input("Student's Last Name: ")
            course_name = input("Course Name: ")
            csv_data = f"{student_first_name},{student_last_name},"\
                f"{course_name}"
        # Present the Current Data
        case 2:
            if student_first_name == "" or student_last_name == "" or \
                course_name == "":  # Skips operation if none registered
                print("\n---------------------------------------")
                print("!!! Please register a student first !!!")
                print("---------------------------------------")
            else:
                print("\nCurrent Data Input:")
                print(csv_data)
        # Save the Data
        case 3:
            if student_first_name == "" or student_last_name == "" or \
                    course_name == "":  # Skips operation if none registered
                print("\n---------------------------------------")
                print("!!! Please register a student first !!!")
                print("---------------------------------------")
            else:
                file_obj = open(FILE_NAME, "a")
                file_obj.write(csv_data + "\n")
                file_obj.close()
                print("\nStored the Following Data:")
                print(csv_data)
        # Stop the Loop
        case 4:
            print("\n-----------------------------------")
            print("*** Exiting Program. Thank you! ***")
            print("-----------------------------------")
            break
        # Error Processing
```



```
        case _:
            print("\n----------------------------------------------------")
            print("!!! Please choose a menu option (1, 2, 3, or 4). !!!")
            print("----------------------------------------------------")
```

Figure 3: ‘Assignment03’ Python Script

The Python script is a simple command-line course registration program that demonstrates the use of conditional logic and looping in Python. Here’s a summary of its main functionalities:

1. **Menu Display:** The script starts by displaying a menu with four options: registering a student, showing current data, saving data to a file, and exiting the program.
2. **User Input Handling:** The program continuously prompts the user for a menu selection and processes the input using a match statement.
3. **Option 1: Register a Student for a Course:** If the user selects option 1, they can input a student's first name, last name, and the course name. The script constructs a CSV-formatted string (“csv_data”) to hold the student's information.
4. **Option 2: Show Current Data:** If the user selects option 2, the script checks if any student data has been registered. If no data is registered, it prompts the user to register a student first. If data is available, it displays the current student information.
5. **Option 3: Save Data to a File:** If the user selects option 3, the script again checks if student data has been registered. If data exists, it appends the “csv_data” string to a file named ‘Enrollments.csv’. If no data is available, it prompts the user to register a student first.
6. **Option 4: Exit the Program:** If the user selects option 4, the program exits the loop.
7. **Error Handling:** If the user inputs an invalid menu option, the program displays an error message and prompts the user to choose a valid option.

Concepts Demonstrated

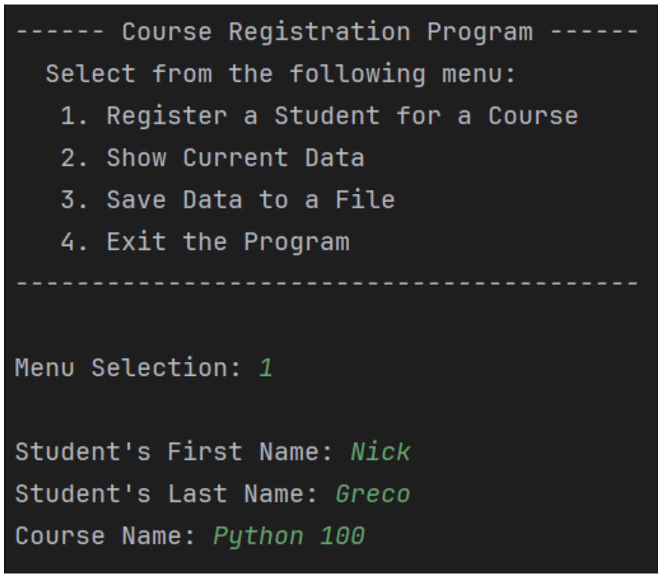
The script demonstrates several concepts from previous assignments, along with the following additions:

1. **Conditional Logic** - The use of the match statement allows for conditional branching based on the user's menu choice. This highlights how to implement control flow in a program based on different conditions.
2. **Looping** - The while loop allows the program to continuously prompt the user for input until they choose to exit. This demonstrates the concept of infinite loops and how to control them with a break statement.
3. **Error Handling** - The script includes checks to ensure that certain operations (like showing or saving data) are only executed if there is registered student information. This preventive logic helps manage user errors gracefully.
4. **Data Validation** - The script checks whether required information (student’s name and course name) is provided before attempting to show or save the data. This highlights the importance of validating user input to ensure the program behaves as expected.

Running the Script

Using PyCharm IDE

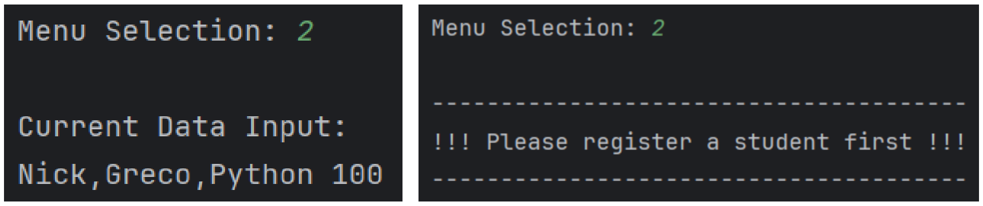
The script was initially run using PyCharm. Figure 4 shows the menu presented to the user and displayed prompts when selection “1” is chosen. Example user input is shown in green text.



```
------ Course Registration Program ------
  Select from the following menu:
   1. Register a Student for a Course
   2. Show Current Data
   3. Save Data to a File
   4. Exit the Program
-----------------------------------------


Menu Selection: 1

Student's First Name: Nick
Student's Last Name: Greco
Course Name: Python 100
```

Figure 4: Menu and Menu Selection “1” Prompts

Figure 5 shows the displayed prompts when menu selection “2” is chosen for scenarios where (left) student information was entered and (right) student information was not entered.

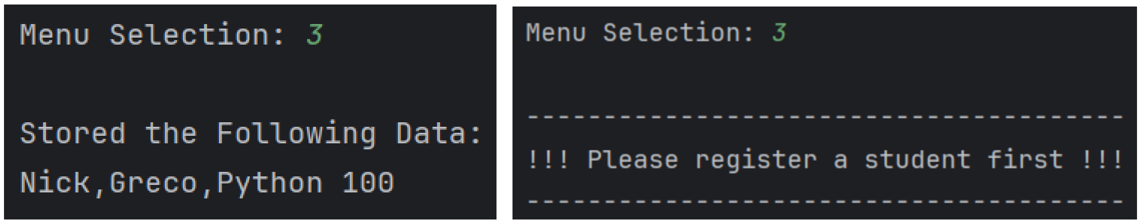


```
Menu Selection: 2

Current Data Input:
Nick,Greco,Python 100
```

```
Menu Selection: 2

--------------------------------------
!!! Please register a student first !!!
--------------------------------------
```

Figure 5: Menu Selection “2” Prompts

Figure 6 shows the displayed prompts when menu selection “3” is chosen for scenarios where (left) student information was entered and (right) student information was not entered.

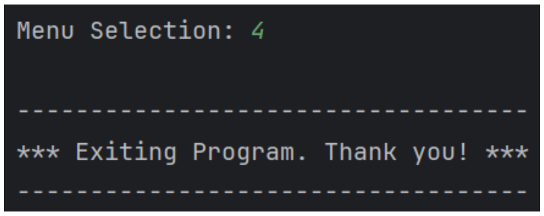


```
Menu Selection: 3

Stored the Following Data:
Nick,Greco,Python 100
```

```
Menu Selection: 3

--------------------------------------
!!! Please register a student first !!!
--------------------------------------
```

Figure 6: Menu Selection “3” Prompts

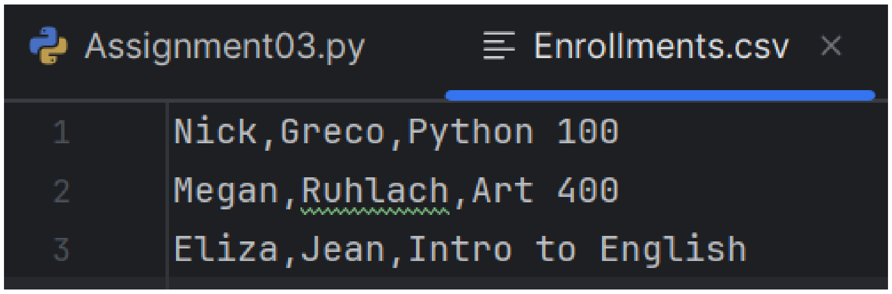
Figure 7 shows the displayed exit message when menu selection “4” is chosen.



```
Menu Selection: 4


----------------------------------
*** Exiting Program. Thank you! ***
----------------------------------
```

Figure 7: Menu Selection “4” Exit Message

Figure 8 shows the resulting ‘Enrollments.csv’ text file where the student registration data is saved after choosing menu selection “3”. In the example shown below, three students’ information was registered through the use of the while loop.

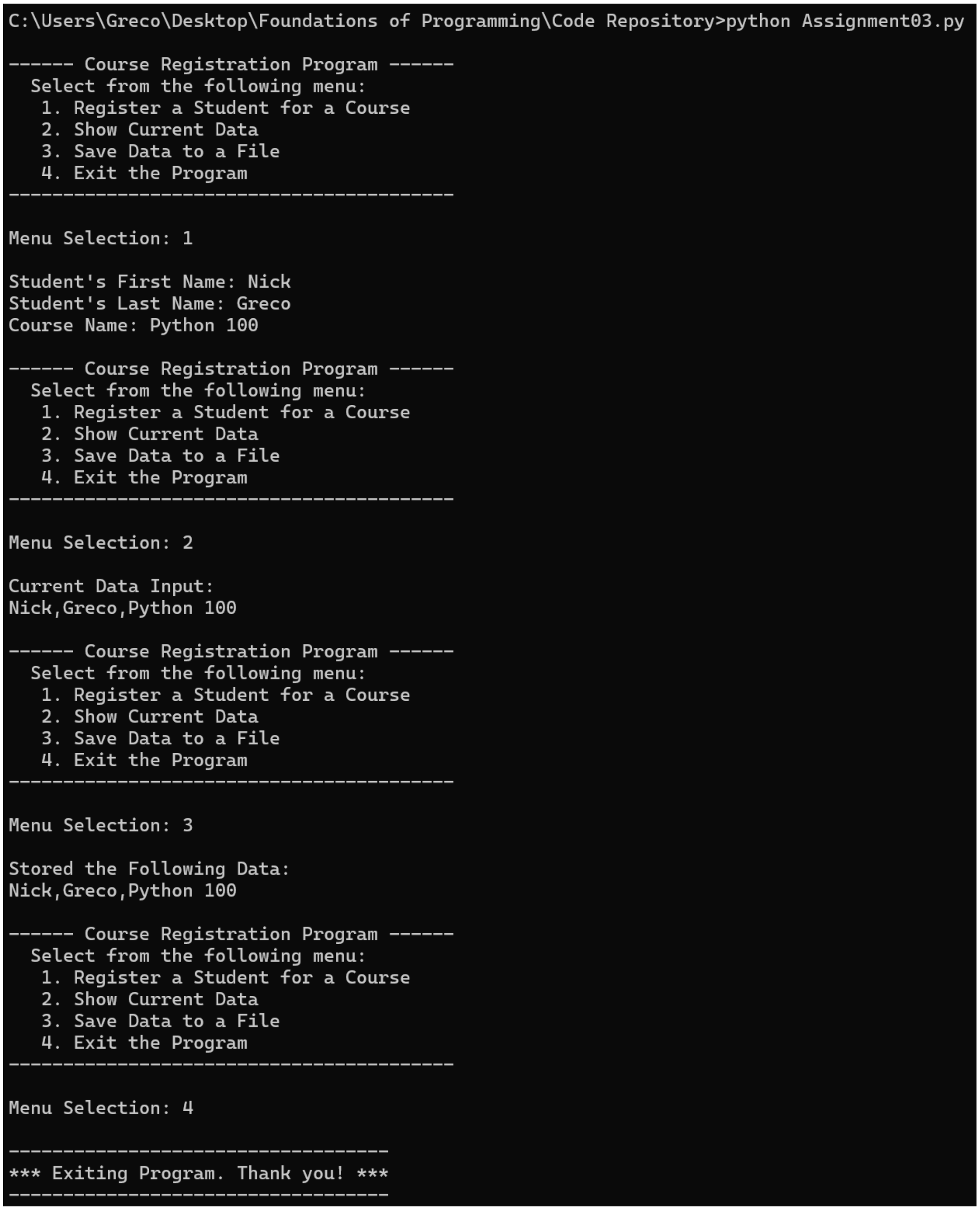


```
Nick,Greco,Python 100
Megan,Ruhlach,Art 400
Eliza,Jean,Intro to English
```

Figure 8: ‘Enrollments.csv’ Data Contents

Using Windows Command Prompt

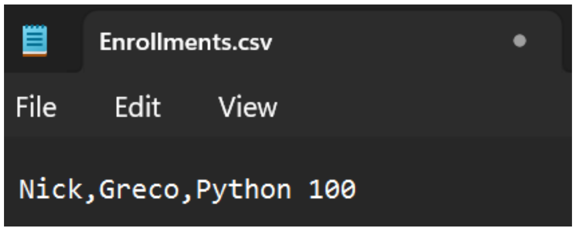
Figure 9 shows running the same script using Windows Command Prompt, but only entering, displaying, and saving a single student’s information.



```
C:\Users\Greco\Desktop\Foundations of Programming\Code Repository>python Assignment03.py

------ Course Registration Program ------
  Select from the following menu:
   1. Register a Student for a Course
   2. Show Current Data
   3. Save Data to a File
   4. Exit the Program
-----------------------------------------

Menu Selection: 1

Student's First Name: Nick
Student's Last Name: Greco
Course Name: Python 100

------ Course Registration Program ------
  Select from the following menu:
   1. Register a Student for a Course
   2. Show Current Data
   3. Save Data to a File
   4. Exit the Program
-----------------------------------------

Menu Selection: 2

Current Data Input:
Nick,Greco,Python 100

------ Course Registration Program ------
  Select from the following menu:
   1. Register a Student for a Course
   2. Show Current Data
   3. Save Data to a File
   4. Exit the Program
-----------------------------------------

Menu Selection: 3

Stored the Following Data:
Nick,Greco,Python 100

------ Course Registration Program ------
  Select from the following menu:
   1. Register a Student for a Course
   2. Show Current Data
   3. Save Data to a File
   4. Exit the Program
-----------------------------------------

Menu Selection: 4

-----------------------------------
*** Exiting Program. Thank you! ***
-----------------------------------
```

Figure 9: Using Windows Command Prompt to run 'Assignment03.py'

Figure 10 shows the contents of the CSV file after executing the script using Command Prompt.



```
Nick,Greco,Python 100
```

Figure 10: CSV File Contents

Summary

This document outlines the steps to create a basic student enrollment system using the PyCharm IDE, including installation and configuration instructions. It demonstrates the use of conditional logic and looping in Python to manage student registrations effectively, allowing users to register students, display current data, and save information to a CSV file. Additionally, it highlights PyCharm's productivity features, such as intelligent code completion and debugging tools, which enhance the development process.

Citations

OpenAI ChatGPT. (October 2024). https://chatgpt.com/: Aspects of this assignment were informed and created by queries I submitted to ChatGPT.