Developers Institute

Python Course

Week 2

Day 3

Mini-Project: Bus Ticket Machine

1. We'll be building a bus ticketing system. Our system should allow a user to:
    1. book a seat on the bus
    2. pay for their ticket
    3. get information about the bus and its seats.
2. Create 5 empty functions: (Use the 'pass' keyword to create a 'placeholder' function body. You'll get back to filling in the function body with real code logic later.)
    1. main() - this will be the main function of our code
    2. book_seat() - when called, this function will allow a user to book a seat on the bus
    3. pay_for_tickets() - when called, this function will allow a user to pay for tickets
    4. show_bus_info() - when called, this function will display information about the bus
    5. show_menu() - when called, this function will display the program's menu.
3. Now that you have the 'skeleton' functions, let's go back and fill them in, one at a time.
4. **main()** function:
    1. This function should:
        1. Display a welcome message
        2. Show the menu (do we already have a function for that…?)
        3. Display a parting message (like 'Goodbye!')
    2. Call this function at the end of your file, so that when you run your file with Python, the main function will be called.
5. **show_menu**() function:
    1. This function should:
        1. Print out the program's interactive menu to the user. Here's an example:
            1. " (a) – book a seat
                (b) – pay for a ticket
                (c) – view information about the bus
                (x) – exit"
        2. Ask the user to select an option, and remember their choice.
        3. Call whatever function matches the user's input, eg. if the user typed 'a', call the **book_seat** function.
        4. But if there is no function that matches the user's input, print an error message.
        5. Show the menu 100 times – but when the user chooses to exit (eg. by typing 'x'), the loop (and also the show_menu() function) should end.
        6. (Bonus: show the menu an infinite amount of times...)
6. **book_seat**() function:

1. Introduction
    1. Our bus has 10 seats. We'll represent this as a list of strings. If a string in the list has a value of 'x', this means that that seat has been booked. If the string is empty (''), it means that that seat is available. For example: ['x', '', '', '', 'x'] → the first and last seats are booked, but the other seats are available.
    2. What should our list be called (as a variable name)? Where should it be initialized within our code?
2. Our **book_seat**() function should do the following:
    1. First, check whether there are any empty seats on the bus. If there are no empty seats, print a message to the user saying so, and end the function.
    2. If there are empty seats (at least one), ask the user to confirm that they want to book a seat. (eg. "Please confirm you want to book a seat on the bus. Type (y) to confirm, or (n) to cancel.")
    3. After the user confirms, change one of the 'available seats' in our string list to a 'booked seat'.
    4. If the user cancels, do not make any changes to the string list.
    5. Now the function should end.
7. **pay_for_ticket**() function:
    1. Introduction
        1. This function will allow the user to 'pay' for one or more tickets that he has already booked.
        2. We will simply assume that the user has actually booked tickets – don't worry about remembering that in this program.
    2. Print a friendly message to the user, telling her how much a ticket costs. Also mention that our machine only accepts bank notes, not coins – 20, 50, 100, and 200 currency notes only.
    3. Ask the user how many tickets he booked. Store this in a variable.
    4. Ask the user to 'insert' bank notes to pay for the tickets. Store the input in a variable.
    5. Examine the input:
        1. If it's too small to cover the cost of the tickets, print an appropriate error message.
        2. If it's not a number that's divisible by 20, 50, 100, or 200, print an error message saying that the user has tried to pay with unacceptable currency.
        3. Otherwise, calculate the change, and 'output' the change to the user.
    6. Print a thank-you message.
    7. The function should now end.
8. **show_bus_info**() function:
    1. Print out some information about the bus (make, model, year, etc.)
    2. Print out how many seats the bus has, and how many of those seats are booked for the bus ride.
9. (BONUS: Add a function called **depart**()
    1. This should 'make the bus leave'.
    2. Of course, we don't have an actual bus. But what should happen is that all of our data about the seats on the bus should be reset! All of the seats on the bus should now become available.)