

## Project 6-1: Prime Number Checker

Create a program that checks whether a number is a prime number and displays its factors if it is not a prime number.

### Console

```
Prime Number Checker

Please enter an integer between 1 and 5000: 5
5 is a prime number.

Try again? (y/n): y

Please enter an integer between 1 and 5000: 6
6 is NOT a prime number.
It has 4 factors: 1 2 3 6

Try again? (y/n): y

Please enter an integer between 1 and 5000: 200
200 is NOT a prime number.
It has 12 factors: 1 2 4 5 8 10 20 25 40 50 100 200

Try again? (y/n): n

Bye!
```

### Specifications

- A prime number is divisible by two factors (1 and itself). For example, 7 is a prime number because it is only divisible by 1 and 7.
- If the user enters an integer that's not between 1 and 5000, the program should display an error message.
- If the number is a prime number, the program should display a message to that effect.
- If the number is not a prime number, the program should display a message to that effect. Then, it should display the number of factors for the number and a list of those factors.
- Store the factors for each number in a list.
- Use functions to organize the code for this program.

## Project 6-2: Wizard Inventory

Create a program that keeps track of the items that a wizard can carry.

### Console

```
The Wizard Inventory program

COMMAND MENU
show - Show all items
grab - Grab an item
edit - Edit an item
drop - Drop an item
exit - Exit program

Command: show
1. wooden staff
2. wizard hat
3. cloth shoes

Command: grab
Name: potion of invisibility
potion of invisibility was added.

Command: grab
You can't carry any more items. Drop something first.

Command: show
1. wooden staff
2. wizard hat
3. cloth shoes
4. potion of invisibility

Command: edit
Number: 1
Updated name: magic wooden staff
Item number 1 was updated.

Command: drop
Number: 3
cloth shoes was dropped.

Command: exit
Bye!
```

### Specifications

- Use a list to store the items. Provide three starting items.
- The wizard can only carry four items at a time.
- For the edit and drop commands, display an error message if the user enters an invalid number for the item.
- When you exit the program, all changes that you made to the inventory are lost.

## Project 6-3: Contact Manager

Create a program that a user can use to manage the primary email address and phone number for a contact.

### Console

```
Contact Manager

COMMAND MENU
list - Display all contacts
view - View a contact
add - Add a contact
del - Delete a contact
exit - Exit program

Command: list
1. Guido van Rossum
2. Eric Idle

Command: view
Number: 2
Name: Eric Idle
Email: eric@ericidle.com
Phone: +44 20 7946 0958

Command: add
Name: Mike Murach
Email: mike@murach.com
Phone: 559-123-4567
Mike Murach was added.

Command: del
Number: 1
Guido van Rossum was deleted.

Command: list
1. Eric Idle
2. Mike Murach

Command: exit
Bye!
```

### Specifications

- Use a list of lists to store the data for the contacts. Provide starting data for two or more contacts.
- For the view and del commands, display an error message if the user enters an invalid contact number.
- When you exit the program, all changes that you made to the contact list are lost.

## Project 6-4: Quarterly Sales

Create a program that gets quarterly sales from a user and calculates the total of all four quarters as well as the average, lowest, and highest quarters.

### Console

```
The Quarterly Sales program

Enter sales for Q1: 12312.57
Enter sales for Q2: 15293.21
Enter sales for Q3: 14920.95
Enter sales for Q4: 23432.21

Total:                65958.94
Average Quarter:      16489.74
Lowest Quarter:       12312.57
Highest Quarter:      23432.21
```

### Specifications

- Use a list to store the sales for each quarter.
- Round the results of each entry to a maximum of 2 decimal digits.

## Project 6-5: Tic Tac Toe

Create a two-player Tic Tac Toe game.

### Console

```
Welcome to Tic Tac Toe

+---+---+---+
|   |   |   |
+---+---+---+
|   |   |   |
+---+---+---+
|   |   |   |
+---+---+---+

X's turn
Pick a row (1, 2, 3): 1
Pick a column (1, 2, 3): 1

+---+---+---+
| X |   |   |
+---+---+---+
|   |   |   |
+---+---+---+
|   |   |   |
+---+---+---+

O's turn
Pick a row (1, 2, 3): 1
Pick a column (1, 2, 3): 2

...

...

X's turn
Pick a row (1, 2, 3): 3
Pick a column (1, 2, 3): 3

+---+---+---+
| X | O | O |
+---+---+---+
|   | X |   |
+---+---+---+
|   |   | X |
+---+---+---+

X wins!

Game over!
```

### Specifications

- Use a list of lists to store the Tic Tac Toe grid.
- If the user picks an invalid row or column or a cell that's already taken, display an error message.
- If there is a winner, the game should display an appropriate message and end. Otherwise, it should continue until the grid is full and end in a tie.