## Project 8-1: Tip Calculator

Add exception handling to a Tip Calculator program.

### Console

```
Tip Calculator

INPUT
Cost of meal: ten
Must be valid decimal number. Please try again.
Cost of meal: -10
Must be greater than 0. Please try again.
Cost of meal: 52.31
Tip percent:  17.5
Must be valid integer. Please try again.
Tip percent:  20

OUTPUT
Cost of meal: 52.31
Tip percent:  20%
Tip amount:   10.46
Total amount: 62.77
```

### Specifications

- The program should accept decimal entries like 52.31 and 15.5 for the cost of the meal.
- The program should accept integer entries like 15, 20, and 25 for the tip percent.
- The program should validate both user entries. That way, the user can't crash the program by entering invalid data.
- The program should only accept numbers that are greater than 0.
- The program should round results to a maximum of two decimal places.

## Project 8-2: Wizard Inventory

Add exception handling to a program that keeps track of the inventory of items that a wizard can carry. If you've done project 7-2, you can add the exception handling to that program. Otherwise, you can start this program from scratch.

### Console if the program can't find the inventory file

```
The Wizard Inventory program

COMMAND MENU
walk - Walk down the path
show - Show all items
drop - Drop an item
exit - Exit program

Could not find inventory file!
Wizard is starting with no inventory.

Command: walk
While walking down a path, you see a crossbow.
Do you want to grab it? (y/n): y
You picked up a crossbow.

Command: show
1. a crossbow

Command: drop
Number: x
Invalid item number.

Command:
```

### The error message if the program can't find the items file

```
Could not find items file.
Exiting program. Bye!
```

### Specifications

- This program should read the text file named wizard_all_items.txt that contains all the items a wizard can carry. Your instructor should provide this file if you don't already have it.
- When the user selects the walk command, the program should randomly pick one of the items that were read from the text file that the user hasn't already grabbed and give the user the option to grab it.
- The current items that the wizard is carrying should be saved in an inventory file. Make sure to update this file every time the user grabs or drops an item.
- The wizard can only carry four items at a time, and those items must be different. For the drop command, display an error message if the user enters an invalid integer or an integer that doesn't correspond with an item.
- Handle all exceptions that might occur so the user can't cause the program to crash. If the all items file is missing, display an appropriate error message and exit the program.
- If the inventory file is missing, display an appropriate error message and continue with an empty inventory for the wizard. That way, the program will write a new inventory file when the user adds items to the inventory.

## Project 8-3: Contact Manager

Add exception handling to a program that manages the primary email address and phone number for a contact. If you've done project 7-3, you can add the exception handling to that program. Otherwise, you can start this program from scratch.

### Console if the contacts file is not found

```
Contact Manager

Could not find contacts file!
Starting new contacts file...

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

Command: list
There are no contacts in the list.

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

Command: list
1. Mike Murach

Command: view
Number: 2
Invalid contact number.

Command: view
Number: x
Invalid integer.

Command: view
Number: 1
Name: Mike Murach
Email: mike@murach.com
Phone: 559-123-4567

Command: exit
Bye!
```

### Specifications

- When the program starts, it should read the contacts from a CSV file named contacts.csv. Your instructor should provide this file if you don't already have it.
- If the program can't find the CSV file, it should display an appropriate message and create a new CSV file that doesn't contain any contact data.
- For the view and del commands, display an appropriate error message if the user enters an invalid integer or an invalid contact number.
- When you add or delete a contact, the change should be saved to the CSV file immediately. That way, no changes are lost, even if the program crashes later.

## Project 8-4: Monthly Sales

Add exception handling to a program that reads the sales for 12 months from a file and calculates the total yearly sales as well as the average monthly sales. If you've done project 7-4, you can add the exception handling to that program. Otherwise, you can start this program from scratch.

### Console

```
Monthly Sales program

COMMAND MENU
monthly - View monthly sales
yearly  - View yearly summary
edit    - Edit sales for a month
exit    - Exit program

Command: edit
Three-letter Month: Dec
Sales Amount: TK
Sales amount for Dec was modified.

Command: monthly
Jan - 14317
Feb - 3903
Mar - 1073
Apr - 3463
May - 2429
Jun - 4324
Jul - 9762
Aug - 15578
Sep - 2437
Oct - 6735
Nov - 88
Dec - TK

Command: yearly
Using sales amount of 0 for Dec.
Yearly total:     64109
Monthly average:  5342.42

Command: exit
Bye!
```

### Specifications

- When the program starts, it should read the sales data from a CSV file named monthly_sales.csv. Your instructor should provide this file if you don't already have it.
- If the program can't find the CSV file when it starts, display an error message and exit the program.
- For the edit command, display an error message if the user doesn't enter a valid three-letter abbreviation for the month.
- When the user edits the sales amount for a month, the data should be saved to the CSV file immediately. That way, no data is lost, even if the program crashes later.
- If the CSV file doesn't contain a valid integer for the sales amount for the month, use a value of 0 to calculate the total sales for the year.
- Round the results of the monthly average to a maximum of 2 decimal digits.