# CS50P - Problem Set 3

## Fuel Gauge

* Implement a program that prompts the user for a **fraction**, formatted as **X/Y**, wherein each of **X** and **Y** is an **integer**, and then outputs, as a **percentage** **rounded to the nearest integer**, how much fuel is in the tank.
* If X or Y is not an integer, X is greater than Y, or Y is 0, instead prompt the user again
* Catch **ValueError** and **ZeroDivisionError**
* If **1% or less** remains, output **E**
* If **99% or more** remains, output **F**
* num, deno = fraction.split(“**/**”)
* print(f“{**percent(num, deno)**}”)
* **except** (ValueError, ZeroDivisionError):
* return str(prct) **+** “%”

## Felipe’s Taqueria

* Implement a program that enables a user to place an order, **prompting them for items**, **one per line**, **until the user inputs control-d** (which is a common way of ending one’s input to a program).
* After each inputted item, **display the total cost of all items inputted this far**, prefixed with a dollar sign (**$**) and formatted to **two decimal places.**
* Treat the user’s input **case insensitively**
* **Ignore** any input that isn’t an item.
* Assume that every item on the menu will be **titlecased**
* Dictionary values can be str, int, float, etc.
* total = 0

**if** order **in** tacos:

total **+=** tacos**[**order**]**

print(**f**”Total: ${total**:.2f**}”)

**else**:

continue

* except EOFError

## 

## Grocery List

* Implement a program that prompts the user for items, **one per line**, until the user inputs **ctrl-d**.
* Then output the user’s grocery list in all **UPPERCASE**
* **Sorted alphabetically** by item
* Prefixing each line with the number of times the user inputted that item. No need to pluralize the items.
* Treat the user’s input case-insensitively.
* items\_list = [ ]

final\_items = { }

* items\_list.append(item)
* **except** EOFError
* sorted(items\_list)
* **for** item **in** items\_list:

**if** item **in** final\_items:

final\_items[item] += 1

**else**:

final\_items[item] = 1

* **for** item **in** final\_items.keys():

print(**f**"{final\_items**[**item**]**} {item}")

## Outdated

* Implement a program that **prompts** the user for a date in month-day-year order, formatted like **9/8/1636** or **September 8, 1636**
* Then **output** the same date i**n YYYY-MM-DD** format.
* If the user’s input is **not a valid dat**e in either format, **prompt the user again**.
* Assume that **every month has no more than 31 days**; no need to validate whether a month has 28, 29, 30 or 31 days.
* If days > 31 = **reprompt**
* If months > 12 = **reprompt**
* If date is in wrong format = **reject**
* **except** ValueError
* list.index(x) # Returns index of first occurrence of x
* split(“**/**”)
* print(f“{year}-{month:02}-{day:02}”) # n:02 notation adds 0 prefix if single integer
* date = date.replace(“**,** “, “”) # Replaces “,” by nothing
* date\_parts = date.split(“ ”) # Splits string from whitespace and creates a list of elements
* month, day, year = date\_parts # Renames date\_parts[0] month, date\_parts[1] day and date\_parts[2] year
* months.index(month) # Returns index of element