Midterm Programming Exam Question Bank

The purpose of the midterm programming exam is to test your ability to create code on your own using minimal help documentation. This may be different than how you have approached your homework problems and lab problems, so it is important that you practice before the exam.

Please practice for the midterm programming exam by using only IDLE and the first page of the two-page reference help sheet. (Previous programming problems, the Internet, and AI are not allowed to help to ensure you understand what you are doing.)

Some programming problems below will be easier to do than others. I will use similar-style problems on the midterm. Aim to spend 15 minutes on each problem on average.

Basic IPO (Input – Process – Output):

Ticket Price Calculator: Create a program that calculates the total ticket price for a movie based on the number of children, adults, and seniors. The ticket prices are as follows:

- Children: $5
- Adults: $10
- Seniors: $7

The program should prompt for the number of children, number of adults, and number of seniors, in that order. It should then calculate the total ticket price. For example:

```
Enter the number of children: 1
Enter the number of adults: 2
Enter the number of seniors: 1
The total price is 32 dollars.
```

How Many Quarters: Create a program that asks for the amount of money to convert to quarters. It should then print how many whole quarters are in the amount of money.

```
Enter the amount of money: 1.75
The total number of quarters is 6
```

Here is another example:

```
Enter the amount of money: 2.01
The total number of quarters is 8
```

How Many Not Quarters: Create a program that asks for the amount of money to convert to quarters. It should then print how much money (in pennies) that could not be converted into whole quarters. For example:

```
Enter the amount of money: 2.31

Total pennies needed after quarters are taken out: 6
```

Branching:

Grading System: Write a program that takes a student's score as input and prints out their corresponding grade. Use the following grading scale:

- A: 90-100
- B: 80-89
- C: 70-79
- D: 60-69
- F: Below 60

Additionally, if the student's score is above 95%, print "Excellent!" For example:

```
Please enter the student's score: 71

C
```

Here is another example:

```
Please enter the student's score: 96

A

Excellent!
```

Ticket Price Calculator: Create a program that calculates the ticket price for a movie based on the age of the viewer. The ticket prices are as follows:

- Children (age 0-12): $5
- Adults (age 13-64): $10
- Seniors (age 65 and above): $7

The program should prompt for the age of the customer. It should then print the ticket price for that customer. For example:

```
Enter the age of the customer: 30

The total price is 10 dollars.
```

Leap Year: Write a program that takes a year as input and determines whether it is a leap year or not. A leap year is divisible by 4, but not by 100 unless it is divisible by 400. For example:

```
Enter the year to check: 2024
Leap year!
```

Here is another example:

```
Enter the year to check: 2023
Not a leap year.
```

Problem: Movie Night Decision You and your friends are deciding what movie to watch. Write a program that asks if it's raining, if it's a weekend, and if everyone is available. You will watch a movie only if:

- It's not raining, and it's a weekend, or
- It's not a weekend, and everyone is available.

You may assume answers to input will only be lowercase 'yes' or 'no'. Here is an example:

```
Is it raining: yes
Is it a weekend: no
Is everyone available: no
We will NOT watch a movie.
```

Here is another example:

```
Is it raining: yes
Is it a weekend: no
Is everyone available: yes
We will watch a movie.
```

While sentinel-controlled loop:

Number in Range: Write a program that will prompt the user for a number between 1 and 10 (inclusive). If the user does not enter a number within the correct range, it should prompt again. This should continue to occur until the user finally enters a number in the correct range. Then the program should print 'Correct!' and end. For example:

```
Enter a number in range: -8
Enter a number in range: 999
Enter a number in range: 9
Correct!
```

Sum of Numbers: Write a program that asks the user to enter positive numbers one by one. The program should keep prompting the user for input until a negative number is entered. Calculate and print the sum of all the positive numbers entered. For example:

```
Enter a number: 5
Enter a number: 10
Enter a number: 15
Enter a number: -1
The sum of positive numbers is 30
```

Average of Numbers: Write a program that asks the user to enter positive numbers one by one. The program should keep prompting the user for input until a negative number is entered. Calculate and print the average of all the positive numbers entered.

```
Enter a number: 5
Enter a number: 10
Enter a number: 15
Enter a number: -1
The average of positive numbers is 10
```

Elevator Limit: A freight elevator has a weight limit of 5000lbs. Write a program that ask the user the weight of a new package to place on the elevator in lbs. The program should print 'You may place another package' and repeat asking for the weight of the next package if the total weight is at or below the limit. The program should print 'Do not add this package' and quit if the total weight is above the weight limit. For example:

```
Enter the weight of a package: 4000
You may place another package
Enter the weight of a package: 999
You may place another package
Enter the weight of a package: 2
Do not add this package.
```

Single for/while count-controlled loop:

Sum of Squares: Write a program that calculates the sum of the squares of numbers from 1 to a user-specified limit. Prompt the user to enter the limit. Note, you may not use the sum() function to solve this problem. You may assume the limit will be greater than or equal to 1.

```
Enter the limit: 3
The sum is 14
```

Sum of Numbers: Write a program that calculates the sum of the numbers from 1 to a user-specified limit. Prompt the user to enter the limit. Note, you may not use the sum() function to solve this problem. You may assume the limit will be greater than or equal to 1.

```
Enter the limit: 3
The sum is 6
```

Sum of Even Numbers: Write a program that calculates the sum of the even numbers from 2 to a user-specified limit. The limit may be even or odd. Prompt the user to enter the limit. Note, you may not use the sum() function to solve this problem. You may assume the limit will be greater than or equal to 2.

```
Enter the limit: 6
The sum is 12
```

Strings:

Lowercase Letter: Write a program that will prompt the user for a lowercase letter between ‘a’ and ‘z’ (inclusive). If the user does not enter a letter within the correct range, it should prompt again. This should continue to occur until the user finally enters a letter in the correct range. Then the program should print ‘Correct!’ and end.

```
Enter a letter: A
Enter a letter: 9
Enter a letter: !
Enter a letter: b
Correct!
```

Counting Vowels: Write a program that prompts the user to enter a word. Count and print the number of vowels in the word using a for loop.

```
Please enter a sentence: A weird sentence.
There are 6 vowels.
```

Average Word Length: Write a program that asks the user to enter one word at a time. The program should keep prompting for input until the word 'quit' is entered. Then the program should print the average word length of all words entered (excluding quit). For example:

```
Enter a word: hi
Enter a word: there
Enter a word: buddy
Enter a word: quit
The average word length is 4
```

Get Email Domains: Write a program to prompt the user for an email string. The program should then print the domain name (part after the @ sign). You may assume only valid emails are entered. For example:

```
Enter email: tom.thumb@donuts.com
Domain is donuts.com
```

Nested loops:

Hollow Square Pattern: Use nested loop to implement a program that prints a hollow square pattern using nested loops. Allow the user to specify the size of the square. You may assume that the size of the square will be >= 3 and be odd.

For example:

```
Enter size: 5
* * * * *
*       *
*       *
*       *
* * * * *
```

Backwards Square: Write a program using nested loops that prompts for the square size. It should then print square number of rows of numbers, counting backwards from the square size number to 1. For example:

```
Enter the size of the square: 3
3 2 1
3 2 1
3 2 1
```