Carnegie Mellon University in Qatar

Principles of Computing

15-110 - Fall 2018

Homework Assignment 4

Out: October 25, 2018

Due: November 04, 2018

6pts 1. Write a Python program that inputs a string that contains *only* 0s and 1s (i.e., a binary integer) and prints its decimal equivalent. Submit this program in a separate Python module named `Binary-to-Decimal.py` [Hint: Use the remainder and division operators to pick off the binary number's digits one at a time, from right to left. In the decimal number system, the rightmost digit has a positional value of 1 and the next digit to the left a positional value of 10, then 100, then 1000, and so on. The decimal number 234 can be interpreted as $4 \times 1 + 3 \times 10 + 2 \times 100$. In the binary number system, the rightmost digit has a positional value of 1, the next digit to the left a positional value of 2, then 4, then 8, and so on. The decimal equivalent of binary 1101 is $1 \times 1 + 0 \times 2 + 1 \times 4 + 1 \times 8$, or 1 + 0 + 4 + 8 or, 13.]

6pts 2. A Caesar cipher is a simple substitution cipher based on the idea of shifting each letter of the plain text message a fixed number (called the *key*) of positions in the alphabet. For example, if the key value is 2, the word "Sourpuss" would be encoded as "Uqwtrwuu". The original message can be recovered by re-encoding it using the negative of the key.

Write a Python program that can encode Caesar ciphers. The input to the program will be a string of plain text and the value of the key. The output will be an encoded message where each character in the original message is replaced by shifting it key characters in the Unicode character set. For example, if `c` is a character in the string and key is the amount to shift, then the character that replaces `c` can be calculated as:
`chr(ord(c) + key)`. Submit this program in a separate Python module named `Caesar-Cipher.py`.

8pts 3. Write a Python program that inputs five numbers, each between 10 and 100, inclusive. As each number is read, display it only if it is not a duplicate of a number already read. Use the smallest possible one-dimensional list to solve this problem. Also, display the complete set of unique values input after the user enters each new value. Submit this program in a separate Python module named `Duplicate-Elimination.py`.

15pts 4. As seen in Homework Assignment 3, a prime number is any integer greater than 1 that is evenly divisible only by itself and 1. The *Sieve of Eratosthenes* is a method of finding prime numbers. It operates as follows:

1. Create a list with all elements initialized to True. List elements with prime indices will remain True. All other list elements will eventually be set to False.
2. Starting with list index 2, determine whether a given element is True. If so, loop through the remainder of the list and set to False every element whose index is a multiple of the index for the element with value True. Then continue the process with the next element with value True. For list index 2, all elements beyond element 2 in the list that have indices which are multiples of 2 (indices 4, 6, 8, 10, etc.) will be set to False; for list index 3, all elements beyond element 3 in the list that have indices which are multiples of 3 (indices 6, 9, 12, 15, etc.) will be set to False; and so on.
3. When this process completes, the list elements that are still True indicate that the index is a prime number. These indices can be displayed.

Write a Python program that uses a list of 1000 elements to determine and display the prime numbers between 2 and 999. Ignore list elements 0 and 1. Submit this program in a separate Python module named `Sieve-of-Eratosthenes.py`.

15pts 5. A small airline has just purchased a computer for its new automated reservations system. You have been asked to develop its new system. You are to write an application to assign seats on each flight of the airline's only plane (capacity: 10 seats).

Your application should display the following alternatives: "`Please type 1 for First Class`" and "`Please type 2 for Economy`". If the user types 1, your application should assign a seat in the first-class section (seats 1-5). If the user types 2, your application should assign a seat in the economy section (seats 6-10). Your application should then display a boarding pass indicating the person's seat number and whether it is in the first-class or economy section of the plane.

Use a one-dimensional list to represent the seating chart of the plane. Initialize all the elements of the list to False to indicate that all the seats are empty. As each seat is assigned, set the corresponding element of the list to True to indicate that the seat is no longer available.

Your application should never assign a seat that has already been assigned. When the economy section is full, your application should ask the person if it's acceptable to be placed in the first-class section (and vice versa). If yes, make the appropriate seat assignment. If no, display the message "`Next flight leaves in 3 hours`". Submit this application in a separate Python module named `Airline-Reservation.py`.