CIS7 Project Documentation

Team Name: Anonymous

Case 1
Brandon Muniz
Tanner Van Mill
Case 3
Quy Nguyen
Nadja Hernandez-Vega

Project Information and Details:

Case 1

In case 1, we are tasked to help a solar marketing specialist with navigating the most efficient routes between four different cities on various scales depending on what the user specified. To implement Data Structures theories and concepts, we first assessed how to properly calculate and form the solutions for the user to request and see. The program implements adjacency matrixes and calculator the cheapest path as well as breadth first search. We made the program seamless by introducing a menu and a loop to the menu that keeps the user engaged until finished in a streamlined fashion.

Explanation:

To show all possible route combinations, the program simply prints all possible combinations to the user. For breadth first search, each city was given a string variable then printed to the control panel in the according order. As for the adjecency matrix, a visual table was created to showcase the data for ease of understanding. All options are placed into a do statement with switches and cases.

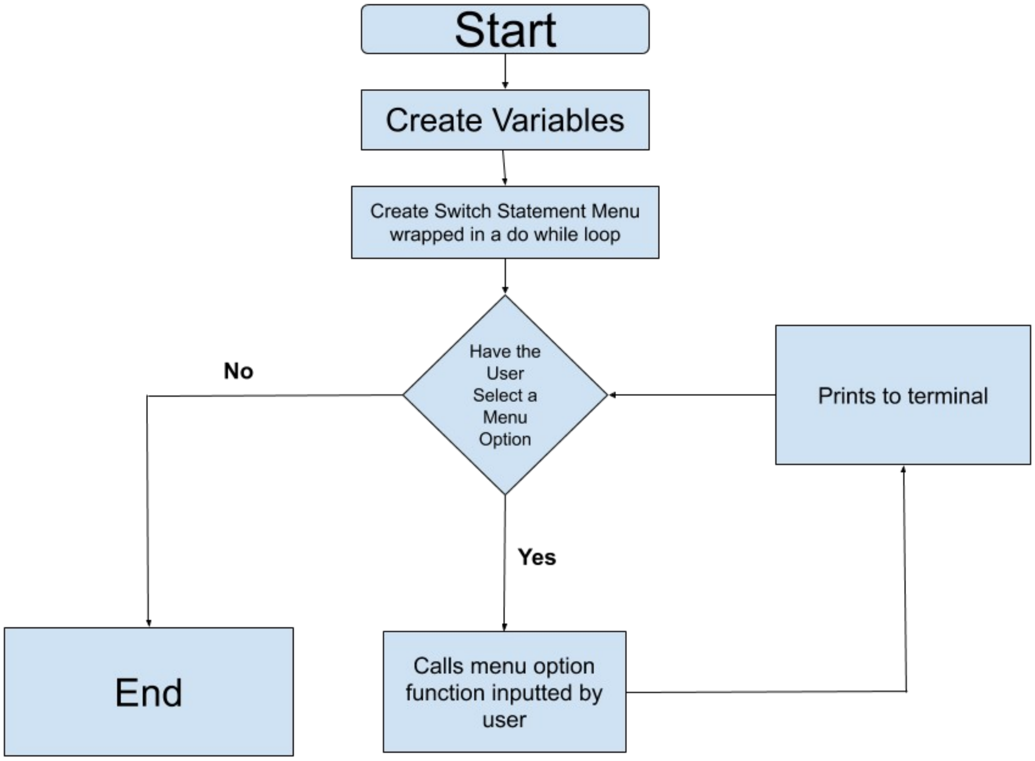
Program limitations include user inputs that break the program. There is no protective measure against character inputs. There is also no preventative measure against inputting large numbers, which also break the program.

A recommendation for absolving the program limitations include creating if statements for every line of code that prompts the user for their input.

Flowchart and Psuedocode:

```
Start Program
Create string variables for the four cities Riverside, Perris, Moreno Valley, and Hemet
Create Integer Variables for the adjacency(roads) between matrices(cities)
Create user input variable (integer) for menu selection
Create a do while statement for the menu
Create switch statement “menu” that prints to terminal
    1. Route Combinations
    2. Breadth First Search
    3. Lowest Mileage Trip
    4. Adjacency Matrix (map and roads)
    5. Exit Program
Take User input to determine users menu choice
```

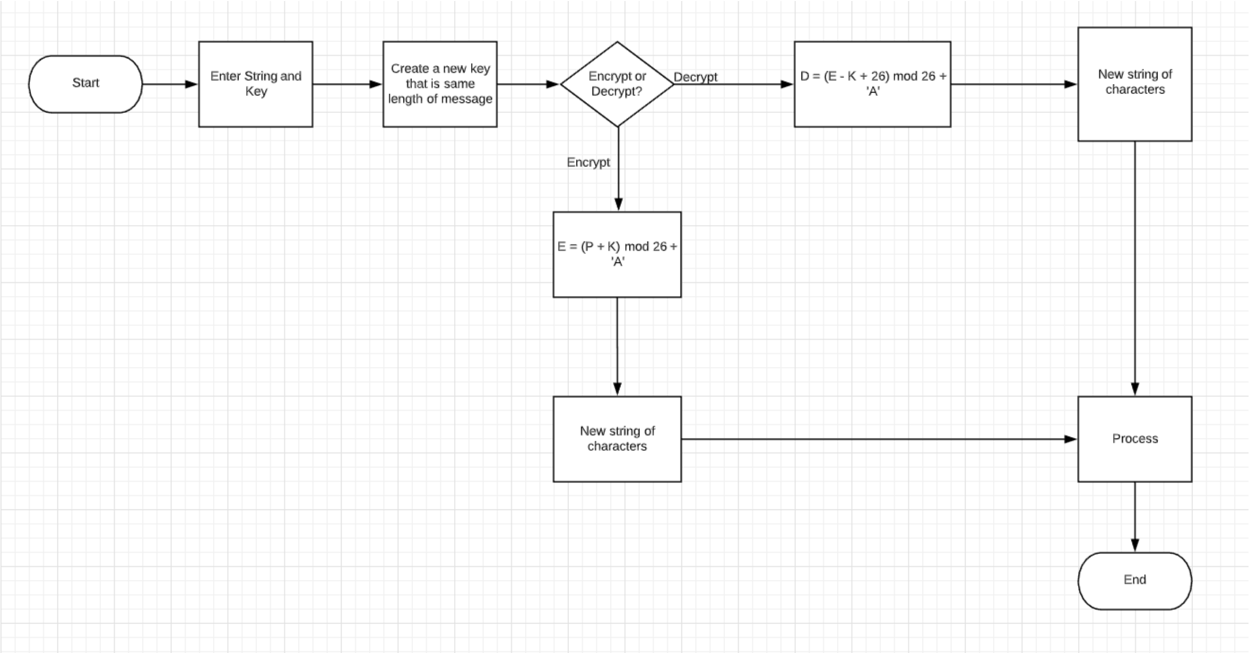
Create Function Calls for each menu option for the menu to print after user inputs console input
Create a function that prints all possible route combinations
Create a function that prints the routes for the breadth first search (see notes ch. 16)
Create a function for shortest path/lowest cost trip (miles)
Shortest path with cities R, P & H
Create a function that prints an adjacency matrix
End Program


Start
Create Variables
Create Switch Statement Menu wrapped in a do while loop
Have the User Select a Menu Option
No
Prints to terminal
Yes
End
Calls menu option function inputted by user

Case 3

1. In this project, we encrypt and decrypt an alphabetic text by Vigenere Cipher method.
2. We will convert the letters to numbers, and then decrypt or encrypt each letter by using a formula using the old text and key. The key must be made the same length as the original message. Then we will switch back to letters using the ASCII table and make a new string with a new message.
3. Explanation:
    - Create a new string key that has the same length as the message by copying each letter in the key until it is the length of the message.
    - In order to encrypt we should change the letters to numbers. Then we can use the formula of: $E = (P + K) \bmod 26$. This formula will calculate which letter is found in the vigenere table.
    - In order to encrypt we should change the letters to numbers. Then we can use the formula: $D = (E - K + 26) \bmod 26$. This formula will find the letter in the vigenere table.
    - Then to convert back to letters, the char A must be added to be found in the ASCII table.
4. The program's objectives are to encrypt or decrypt a message using the Vigenere Cipher method. The program will have a menu that the user can interact with and the menu will continue to loop until the user decides to stop the program. The user will choose whether to encrypt or decrypt. Then the user will input the original or encrypted message depending on which option he chose and then enter the key. Then the program will encrypt or decrypt and then display the new message. Then the user will have the option to continue decrypting or encrypting messages or end the program.
5. Discrete structures are implemented into the program using cryptography and using modular arithmetic. We used the mod to calculate the number of the corresponding letter in the program.
6. A limitation in the program is that the message and key that is imputed from the user cannot be in lowercase letters. It must be uppercase. Another limitation is that there are no spaces allowed either in the message or key that is imputed from the user.
7. To fix these limitations, instead of making them strings, they can be made into vectors of chars. Additionally, we can transform each message into uppercase letters in the program rather than making the user input the message and key in uppercase letters. There cannot be spaces so instead we could have the program not ignore the white space and implement that into the functions to encrypt or decrypt.

3. Flowchart and Pseudocode



Pseudocode:

```
Start
Include iostream
include string

Create new key to ave the same length as the message
        String newKey(string stringToBe, string key)
                Using for loop (int i = 0; ; i++)
                        If size of stringToBe is equal to i
                                i = 0
                        If size of key is equal to size of stringToBe
                                Break
                        Add more letters to the key.
                Return key

Encryption function
        Using for loop
                newMessage += char(((stringToBe[i] + key[i]) % 26) + 'A'))
                Return newMessage

Decryption function
        Using for loop
                newMessage += char((((stringToBe[i] - key[i] + 26) % 26 ) + 'A'))
                Return newMessage

Main
        Variable Declaration
                string stringToBe, key
        Display the menu that asks the user to choose Encrypt, Decrypt or End Program.
        If Encrypt
                Ask the user to input the message and the key
                Generate key length by calling the newKey function
                Encrypt the message by using Encryption function
                Display the Encrypted Message
        If Decrypt
                Ask the user to input the message and the key
                Generate key length by calling the newKey function
                Decrypt the message by using Decryption function
                Display the Decrypted Message
        If End Program
                End Program

End program
```