Overview/Discussion

This program allows a user to process daily high temperatures from a file in a standard format. Each line in the file is of the format: `MonthName, day1High, day2High, …, dayNHigh`

Example:
```
January,32,34,38,45,51,41,39,38,36,33,45,54,56,49,36,35,42,40,47,54,59,36,42,40,36,29,28,33,50,51,34
February,36,42,59,49,37,24,8,9,14,15,15,7,9,4,0,11,22,25,33,42,42,60,69,46,46,52,63,53
```

Other than print, helpful and/or required Python built-in functions and methods include:

`min(list)`	`open(…)`	`listName.index(value)`
`max(list)`	`fileHandle.write(str)`	`range(…)`
`str.endswith(str)`	`fileHandle.close()`	`sys.exit()`
`sum(list)`	`int(str)`	
`len(list`	`input(prompt)`	

Remember operators/keywords such as **in**, **not**, **and**, **or**
You will need **try except**
Formatting FYI:

f"{expression:>10s}" will right-align an expression that is a string in a field width of 10
f"{expression:>10d}" will right-align an expression that is an integer in a field width of 10
f"{expression:<10s}" will left-align an expression that is a string in a field width of 10
f"{expression:<10d}" will left-align an expression that is an integer in a field width of 10

Lists:

months will contain the month names from the file. For example ["January", "February", ...]
dailyHighs is a two-dimensional list (table) where each row is a list of the daily highs for that month. For example,

```
[
  [32,34,38,45,51,41,39,38,36,33,45,54,56,49,36,35,42,40,47,54,59,36,42,40,36,29,28,33,50,51,34],
  [36,42,59,49,37,24,8,9,14,15,15,7,9,4,0,11,22,25,33,42,42,60,69,46,46,52,63,53],
  ...
]
```

Preliminary

- Download and open DailyHighsStudent.py.
- Download and place 2021DailyHighsKCMO.csv and 2021DailyHighsKCMO-Partial.csv in the same folder as DailyHighsStudent.py

Requirements

- **You are permitted to use only the code that we've discussed in the notes or is in the textbook**.
- You may work with up to two other people in the class (3 max in a group). If doing so, write each name after @author and one teammate is to submit one version to Canvas. You must work on the project equally.
- There will be no global variables.
- You are required to follow proper coding style, however, no further internal comments (comments within the functions are required)
- Insert program header documentation – your name as the author and a concise program description. You are permitted to use the first sentence under the Overview/Discussion section above for your description.
- Complete function **main**
  - Complete the remaining code where instructed in the comments. The existing code may not be modified. One extra note for this:
    - For choice "1", assign -1 to a variable named **monthIndex**. **while monthIndex** < 0, prompt for and retrieve the month into a variable named **month**. After the prompt, embed a call to

months.**index**(**month**) inside a **try**. Attach an empty **except** clause (just **except**:) to the **try**. Inside the **except**, print f"{**month**} not recognized". This will continue to loop until a valid index is found.

- Define function **getMenuChoice** directly under its header documentation. The parameter names are to match the documentation.
  - It presents the menu choices as shown in the sample output. There is **<u>no</u>** loop.
  - It prompts the user for the choice and returns the choice (The choice remains a string)
- Define function **getDailyHighs** directly under its header documentation. The parameter names are to match the documentation.
  - Insert a **try except** that catches the object named **Exception**
  - Inside the **try** block
    - Open the file given by the filename parameter.
    - Use one loop to loop through each line (month) in the file. Inside the loop,
      - Append the month name encountered (first value in the line) to the **months** (parameter) list.
      - Insert a loop (nested) to run through the remaining daily highs and append as integers (not strings) onto a list. After this loop, append the list of integers to the **dailyHighs** table
    - Remember to close the file if not using **with**
  - Inside the **except** block,
    - Display the exception caught followed by "Application exit." on the next line
    - If the file handle (e.g. inFile) exists in locals, close it. Then, exit the program.
- Define function **getAverageHigh** directly under its header documentation. The parameter names are to match the documentation.
  - The **monthIndex** parameter is the row index for the **dailyHighs** table parameter. It is the month for which the average is computed → **dailyHighs**[**monthIndex**]
  - Used the built-in <u>sum</u> and <u>len</u> functions to compute and return the average of the daily highs for the given month.
  - This function could be written with only one return statement.
- Define function **showTempsInRange** directly under its header documentation. The parameter names are to match the documentation. There is nothing returned.
  - Display an output header:
    - First display "Day" left-aligned in a field width of 10
    - Use a loop to display the numbers 1 – 31 on the same line as Day. Each number is right-aligned in a field width of 3.
    - After the loop, use print() to advance output to the next line
  - Insert a nested loop to loop through each row in **dailyHighs**:
    - Inside the outside loop, use the index to display the current month from the **months** list
    - Inside the inner loop, if the temperature is within the lower and upper bound (inclusive), display the temperature right aligned in a field width of 3; otherwise display a * (asterisk) right-aligned in a field width of 3.
    - Remember to advance output to the next line after the inner loop concludes.
- Define function **getOverUnder** directly under its header documentation. The parameter names are to match the documentation.
  - Use one nested loop to loop through the **dailyHighs** list and compute three totals: the number of temperatures under the **overUnder** parameter, the number of temperatures equal to **overUnder**, and the number of temperatures above **overUnder**
  - Return a tuple of three integers: (totalUnder,totalEqual,totalOver)
- Complete function **createWebPageStats** directly under its header documentation. The parameter names are to match the documentation.
  - **ROW_COLORS** is a tuple that contains the colors to alternate between for table rows of the months.
  - Follow the instructions and the code in the function.

- The key will be to insert one loop (not nested) that writes data for each table row in an html format. Each table row will be a string of the form:

```
        f'  <tr style = "background-color:{color};text-align:center;">\n' + \
        f'    <td>{months[i]}</td>\n' + \
        f'    <td>{lowestDailyHigh}</td>\n' + \
        f'    <td>{highestDailyHigh}</td>\n' + \
        '  </tr>\n'
```

Where

- **color** is a color that alternates between index 0 and index 1 from the **ROW_COLORS** tuple.
- **i** is the index of the for loop going through each row of **dailyHighs**
- **lowestDailyHigh** will conntain the minimum value from the row currently being examined (use the min function)
- **highestDailyHigh** will contain the maximum value from the row currently being examined (use the max function)

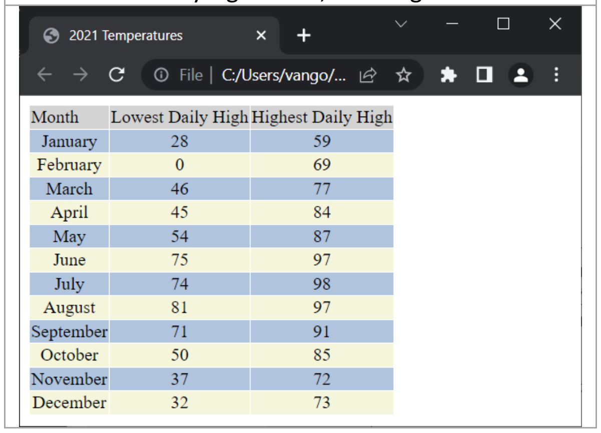
Sample Runs (User input shown in red)

Run 1

```
(1) Show average daily high for a given month.
(2) Show daily high temperatures within a given range.
(3) Show over/under statistics for a given temperature.
(4) Store statistics as Web page.
(5) Exit.
Choice? 1

Month? Setember
Setember not recognized.
Month? Sept
Sept not recognized.
Month? September
Average High: 82.7F

(1) Show average daily high for a given month.
(2) Show daily high temperatures within a given range.
(3) Show over/under statistics for a given temperature.
(4) Store statistics as Web page.
(5) Exit.
Choice? 1

Month? January
Average High: 41.4F

(1) Show average daily high for a given month.
(2) Show daily high temperatures within a given range.
(3) Show over/under statistics for a given temperature.
(4) Store statistics as Web page.
(5) Exit.
Choice? 2

Show temperatures within what range?
Lower bound: 30
Upper bound: 70
Day         1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
January    32 34 38 45 51 41 39 38 36 33 45 54 56 49 36 35 42 40 47 54 59 36 42 40 36  *  * 33 50 51 34
February   36 42 59 49 37  *  *  *  *  *  *  *  *  *  *  *  *  * 33 42 42 60 69 46 46 52 63 53
March      55 62  *  * 57 66 69  *  *  * 60 50 51 50 52 49 61 49 57 63 68 66 59 53 46 62 59 60  * 63 53
April      52 69  *  *  *  *  * 52  * 59  * 61 61 58 60 52 58 61 53 45 49 58 57 67  *  *  *  *  *  *
May         *  * 68 65 69 65  *  * 54 64 62 64 69  * 69 67 69  *  *  *  *  *  *  *  *  *  * 61 67 67 64
June        *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
July        *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
August      *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
September   *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
October     *  *  *  *  *  *  *  *  *  * 65  *  * 70 64 65  *  *  * 68 59 65 59  * 50 60 59 52 55 64 56
November   44 48 50 52 57 65  * 70 66 68 55 43 46 53 65  * 65 48 50 55 58 48 66 60 37 54 67 51 65 55
```

```
December   64  * 65 51 64 36 41 51 63 66 47 56 56 65  * 46 43 32 37 46 45 52 59  * 56 65 62 56 33 43 54

(1) Show average daily high for a given month.
(2) Show daily high temperatures within a given range.
(3) Show over/under statistics for a given temperature.
(4) Store statistics as Web page.
(5) Exit.
Choice? 3

Over/Under temperature? 60
There are 127 temperatures below, 6 equal to, and 232 above 60F.

(1) Show average daily high for a given month.
(2) Show daily high temperatures within a given range.
(3) Show over/under statistics for a given temperature.
(4) Store statistics as Web page.
(5) Exit.
Choice? 4

Name of file to create? 2021DailyHighs
2021DailyHighs.html created.

(1) Show average daily high for a given month.
(2) Show daily high temperatures within a given range.
(3) Show over/under statistics for a given temperature.
(4) Store statistics as Web page.
(5) Exit.
Choice? 4

Name of file to create? 2021Highs.html
2021Highs.html created.

(1) Show average daily high for a given month.
(2) Show daily high temperatures within a given range.
(3) Show over/under statistics for a given temperature.
(4) Store statistics as Web page.
(5) Exit.
Choice? 5
```

2021DailyHighs.html, 2021Highs.html



Month	Lowest Daily High	Highest Daily High
January	28	59
February	0	69
March	46	77
April	45	84
May	54	87
June	75	97
July	74	98
August	81	97
September	71	91
October	50	85
November	37	72
December	32	73

Run 2 – Attempting to open a file that didn't exist

```
[Errno 2] No such file or directory: '2021DailyHighsKC.csv'
Application exit.
```

Run 3 – Encountering an invalid integer temperature

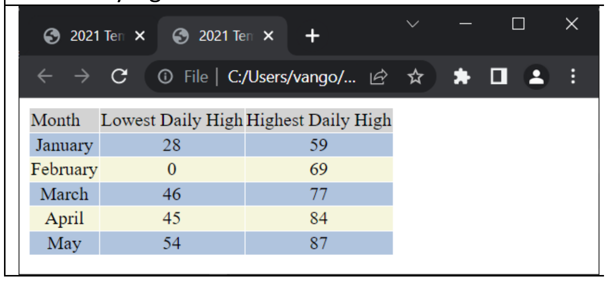
```
invalid literal for int() with base 10: '42a'
Application exit.
```

Run 4 – Using 2021DailyHighsKCMO-Partial.csv

```
(1) Show average daily high for a given month.
(2) Show daily high temperatures within a given range.
(3) Show over/under statistics for a given temperature.
(4) Store statistics as Web page.
(5) Exit.
Choice? 2

Show temperatures within what range?
Lower bound: 0
Upper bound: 60
Day        1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
January   32 34 38 45 51 41 39 38 36 33 45 54 56 49 36 35 42 40 47 54 59 36 42 40 36 29 28 33 50 51 34
February  36 42 59 49 37 24  8  9 14 15 15  7  9  4  0 11 22 25 33 42 42 60  * 46 46 52  * 53
March     55  *  *  * 57  *  *  *  *  * 60 50 51 50 52 49  * 49 57  *  *  * 59 53 46  * 59 60  *  * 53
April     52  *  *  *  *  *  * 52  * 59  *  *  * 58 60 52 58  * 53 45 49 58 57  *  *  *  *  *  *  *
May        *  *  *  *  *  *  *  * 54  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *

(1) Show average daily high for a given month.
(2) Show daily high temperatures within a given range.
(3) Show over/under statistics for a given temperature.
```

```
(4) Store statistics as Web page.
(5) Exit.
Choice? 1

Month? May
Average High: 71.2F

(1) Show average daily high for a given month.
(2) Show daily high temperatures within a given range.
(3) Show over/under statistics for a given temperature.
(4) Store statistics as Web page.
(5) Exit.
Choice? 4

Name of file to create? 2021DailyHighsPartial
2021DailyHighsPartial.html created.

(1) Show average daily high for a given month.
(2) Show daily high temperatures within a given range.
(3) Show over/under statistics for a given temperature.
(4) Store statistics as Web page.
(5) Exit.
Choice? a


(1) Show average daily high for a given month.
(2) Show daily high temperatures within a given range.
(3) Show over/under statistics for a given temperature.
(4) Store statistics as Web page.
(5) Exit.
Choice? 6


(1) Show average daily high for a given month.
(2) Show daily high temperatures within a given range.
(3) Show over/under statistics for a given temperature.
(4) Store statistics as Web page.
(5) Exit.
Choice? 5
```

2021DailyHighsPartial.html



Month	Lowest Daily High	Highest Daily High
January	28	59
February	0	69
March	46	77
April	45	84
May	54	87

Submission

Submit the source code before the due date/time.