C++ Test Documentation for each function

13 functions have been used

`bool is_upper(char input_char)` -> checks if the input character is in Uppercase or not, if it isn't return False, if it is return True

Input	Expected output	Actual output	Passed?
A	1	1	✓
c	0	0	✓
" " [space]	0	0	✓
?	0	0	✓

`bool is_alpha(char input_char)` -> checks if the input character is alpha or not, if it isn't return False, if it is return True

Input	Expected output	Actual output	✓ Passed? or X
A	1		
C	1	1	✓
" "	0	0	✓
$	0	0	✓
c	1	1	✓

`char to_upper(char input_char) -> if the input char is Uppercase then don't do nothing return the character, else if the character isn't Uppercase, then put the character in uppercase`

Input	Expected output	Actual output	Passed?
a	A	A	✓
zdAA	ZDAA	ZDAA	✓
Acccv	ACCCV	A	✓
BravoBE	BRAVOBE	BRAVOBE	✓

`bool is_alphabet(char input_char)  -> function will check if the character is alphabet or not, returning either true if it is alpha character or false if it isn't.`

Input	Expected output	Actual output	Passed?
A	1	1	✓
z	1	1	✓
a	1	1	✓
Z	1	1	✓
" "	0	0	X
£	0	0	X

int max_index(int *array, int size) – prints out the maximum index in the array with biggest number

Input	Expected output	Actual output	Passed?
[1564879]	6[index]	6[index]	✓
[11157863258]	10[index]	10[index]	✓

[aaa]	aaa	aaa	X
" "	" "	" "	X

Lettertype* character_count(vector <char> array, int size);

tests if the character does not exists in the table of characterFrequency then add the letter to the characterFrequency and then count 1. It finds the character in the characterFrequency table, if it exist return it, else return -1

Input in table characterFrequency	Expected output	Actual output	Passed?
helloworld	h➔ 1, L➔1, o➔1, w➔1, r➔1, d➔1	h➔ 1, L➔1, o➔1, w➔1, r➔1, d➔1	✓
programming	m➔ 1	m➔ 1	✓
programming	m➔2	m➔2	✓
helloworld	L➔2,	L➔2,	✓
helloworld	L➔3,	L➔3,	✓
programming helloworld	I➔1 o➔2,	I➔1 o➔2,	✓
helloworld	e➔1	e➔1	✓

- `void decrypt(int key, ifstream& inputFile, ofstream& outputFile)`
- `takes in the key, the encrypted file, decrypts it and writes the output to a file`

0	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
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W

Figure 1.2 , shows a table of alpha characters with an index beginning at 0, the key would be shifted of which a specific letter shows decrypt character, i.e. 'P' with a key of 3 gives a character of 'M'

*the function decrypt will decrypt the string on the encrypted inputFile, using the key found on the function keychalcualtor

* the key to decrypt it and then output the decrypted file into outputFile

Input string (Letter + Key)	Expected output	Actual output	Passed?
"PFT 7" (key = 3)	"MCQ 7"	"MCQ 7 "	✓
HADP (key = 3)	EXAM	EXAM	✓
ZHHN (key = 3)	WEEK	WEEK	✓
V2 (key = 3)	S2	S2	✓
Aol (key = 7)	the	the	✓
Ibpsa (key = 7)	built	built	✓
Puav (key = 7)	into	into	✓

Hss (key = 7)	all	all	✓
Yhaoly (key = 7)	Rather	Rather	✓
, (key = 7)	,	,	✓
! (key = 3)	!	!	✓

- void print_characterFrequency (Lettertype * arrayout, int my_size) – own – prints out character array frequency
- `print character frequency outputs the frequencies of each character in the array by firstly reading the encrypted text in the file`

```
 test @ aaaa ==> 4
test @ helloworld = o==> 2 , l==> 3, h==> 1 , e==> 1, w==> 1 , r==> 1, d==> 1
```

Input	Expected output	Actual output	Passed?
Aaaaa	⇨ 5	⇨ 5	✓
`hello`	`o==> 2 , l==> 3, h==> 1 , e==> 1,`	`o==> 2 , l==> 3, h==> 1 , e==> 1,`	✓
`world`	`w==> 1 , r==> 1, o==> 2 , d➔ 1`	`w==> 1 , r==> 1, o==> 2 , d➔ 1`	✓

Int find_character_in_table(Lettertype* array, int size, char letter_to_search)

* Function to find the character in a table, will do the the following:

* if the letter exists in the table, then return the position of the letter

* if the letter isn't found, -1 will be returned

0	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
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W

Input	Expected output	Actual output	Passed?
hello	H =>	position 7	✓
hello	E=>	position 4	✓
world	a =>	-1	X
lesson	L=>	position 11	✓
week	k=>	position 10	✓
program	b=>	-1	X

vector <char> read_file(string inputFile)

Function to read files,* if the file is open, then open the content in the file

* if for some reason the file can't or isn't opening, then send a consol message

* the while loop will run until one of the two statements is true, i.e. file opened successfully or file did not open successfully.

* Once the file has successfully opened and is alphabetic, then upppercase all the characters and output the frequency of each character

* else if the characters are not alphabetic characters such as !, ? " " , then do nothing.

Input	Expected output	Actual output	Passed?
Inputfile.txt (argv 1)	Inputfile.txt	Main menu	✓
Input.txt (argv 1)	Input.txt	Error occurred reading input file	X
Encrypt.txt (argv 1)	Encrypt.txt	Error occurred reading input file	X
Decrypt.txt (argv 2)	Decrypt.txt	Main menu	✓

void read_decrypted_file(string outputFile) , reads the decrypted file to print out to the user, prints 50 characters

Input	Expected output	Actual output	Passed?
Yhaoly (key = 7)	Rather	Rather	✓
Aohu (key = 7)	than	than	✓
Ylxbpypun (key = 7)	requiring	requiring	✓
Hss (key = 7)	all	all	✓
klzpylk (key = 7)	desired	desired	✓
Mbujapvuhspaf (key = 7)	functionality	functionality	✓
Il (key = 7)	be	be	✓

int keyCalculator(char LanguageFrequentCharacter, char mostFrequentCharacter)

* the function keyCalculator, takes the alphabet position of the letter

* it uses this position to calculate the difference between the two characters

i.e. e and l which gives a key of 7

* test @ e position = 5

* test @ l position = 12

* key = 7

0	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
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W

Abs library is used to prevent results on keycalculator being negative

Input	Expected output	Actual output	Passed?
El (position 5-12)	Key = 7	Key = 7	✓
Ad (position 0-3)	Key = 3	Key = 3	✓

Xb (position 23 -1)	Key = 4	Key = 4	✓
Dp (position 3 -15)	Key = 12	Key = 12	✓

void print_vector(vector <char> arrayout)

* print_vector will print out the array, gives the user information as they go on

Input	Expected output	Actual output	Passed?
programming	mm→ 2	mm→ 2	✓
programming	p→1	p→1	✓
Week ten	e→3	e→3	✓
hello	h→1	h→1	✓