Preparatory data Structure (CSCI 591)

Project - II

Recursive Functions – Three small programs

Submitted By: Taddese Erba

January 28, 2020

St. Cloud state university

Department of Computer Science

Taddese Erba
Section – I
Project – Two
Due: January 30, 2020

Design Document

Introduction

A recursive function is a function that calls itself. By calling itself, a recursive function executes a block of code repeatedly until it hits the smallest possible result(s) also known as the base. Recursive code is important because of its small code size and easiness to write. Once it is written, it can also be used over and over. A recursive function is not always the desired method for solving a problem due to its performance issues. It often takes a long time to compute and return an output than a loop program.

Data Structure

The program uses three data structures, the `class FileStream`, `class Palindrome,` and the function `exp()`. This class `FileStream` holds the character input from a `.dat` file. This class hosts two arrays and three functions as its members. The array `char array[MAX]` is used to hold the characters read in from the file. The second array, `char a[MAX],` is used to hold the array of characters while we search the array for the presence of a character.

The second class is the `class Palindrome.` This class is used to recursively check whether a string entered by the user is a palindrome or not. The class Palindrome has three character type arrays and five-member functions. The array `charArr[MAXLEN]` holds the characters entered by the user. Another array, `arrayToLower[MAXLEN],` is used to hold the characters converted to lower case. The `whiteSpaceRemoved[MAXLEN]` array holds the characters without whitespaces. All the arrays are character data types with their size set to a maximum value.

Taddese Erba
Section – I
Project – Two
Due: January 30, 2020

Functions

There are three functions in the class `FileStream` and five functions used in the class `Palindrome`. The function `void readFile(char [], int)` is used to read the characters from the file one at a time and places these characters in the array `char array[MAX]`. It takes two arguments and returns nothing. The second function, `int getnumOfChars(int)` , is used to count the number of characters read from the file. It takes in integer type variable and returns an integer to the `main` function. The last function in this class is the `bool present(char [], int, int, char)` function that is used to lookup the presence of the character entered by the user. This function takes one character type array, two integer type variables and one character type variable. If it successfully finds or fails to find the character entered by the user, it returns a Boolean result to the `main` function.

The `void getPalindrome (char [], int)` function is used to get the stream of characters from the user. This function takes the `charArr[]` array and another character variable as its arguments and returns no value to the `main` function. Furthermore, this function checks if the user entered the right size of characters and displays a message to the user accordingly. The function `void toLowerCase(char [], int)` takes two arguments, the array `charArr[]` and the array size `int`. It is used to convert the characters from upper to lower case and has no return value. The third function, `void removeWhiteSpaces(char [], int)`, is similar in its structure to the other mentioned functions. Its main purpose is to remove all the whitespaces from the character arrays. The `void reversePalindrome (char [], int)` function is used to write the characters in their reverse order (from last to first). Its structure is similar to the above functions and it returns no value. The last function in this program, the `bool compare (char [], char [], int, int),` is used to compare the original characters entered by the user without the whitespaces to the reversed characters. It has four arguments, `charArr[]`, `arrayReversed []`, `len`, and `len`. The two `len` are the sizes of the two arrays. This function returns a `boolean true` or `false`.

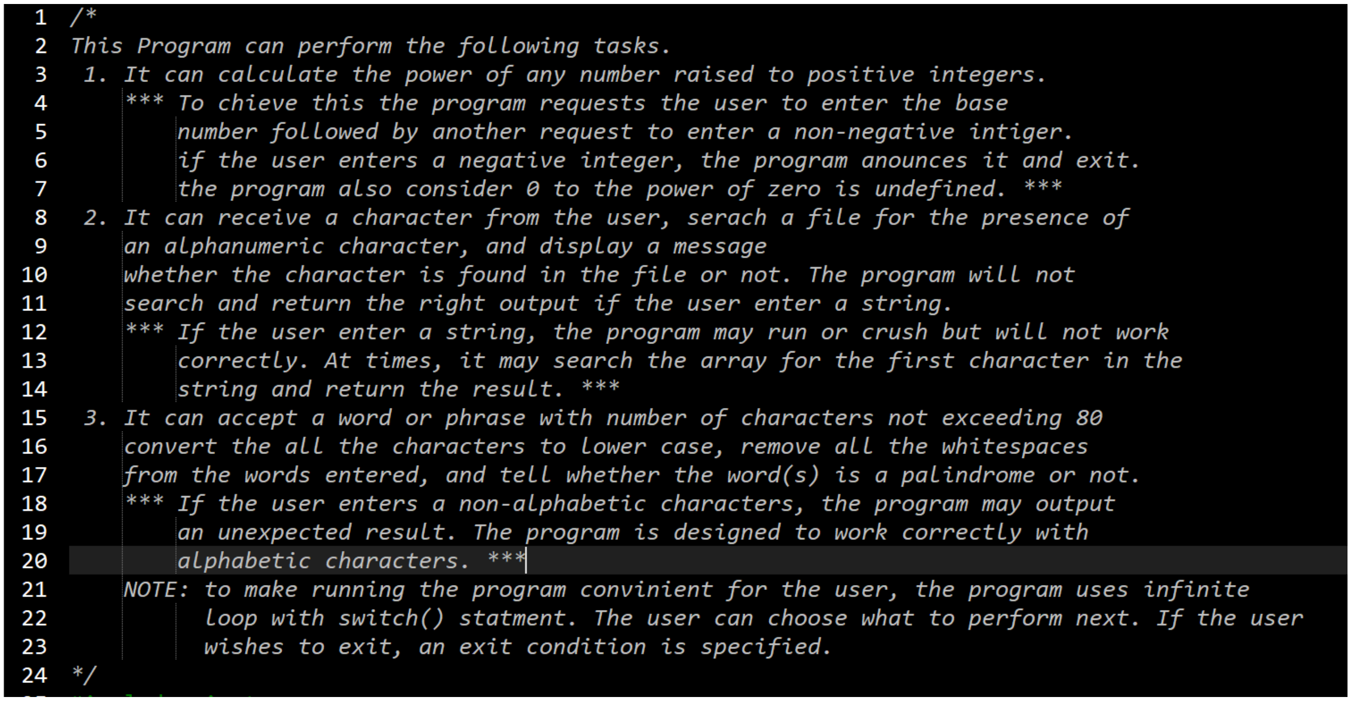
The last function that is used in this program without a class is the `double exp (double, int)` function. This function performs the computation of exponential (power) of any number raised to positive integers. The function is capable of accepting zero and all positive integers as the exponent. The base can be any real number including zero. This program assumes zero to the power of zero is undefined and returns values accordingly.

The Main Program

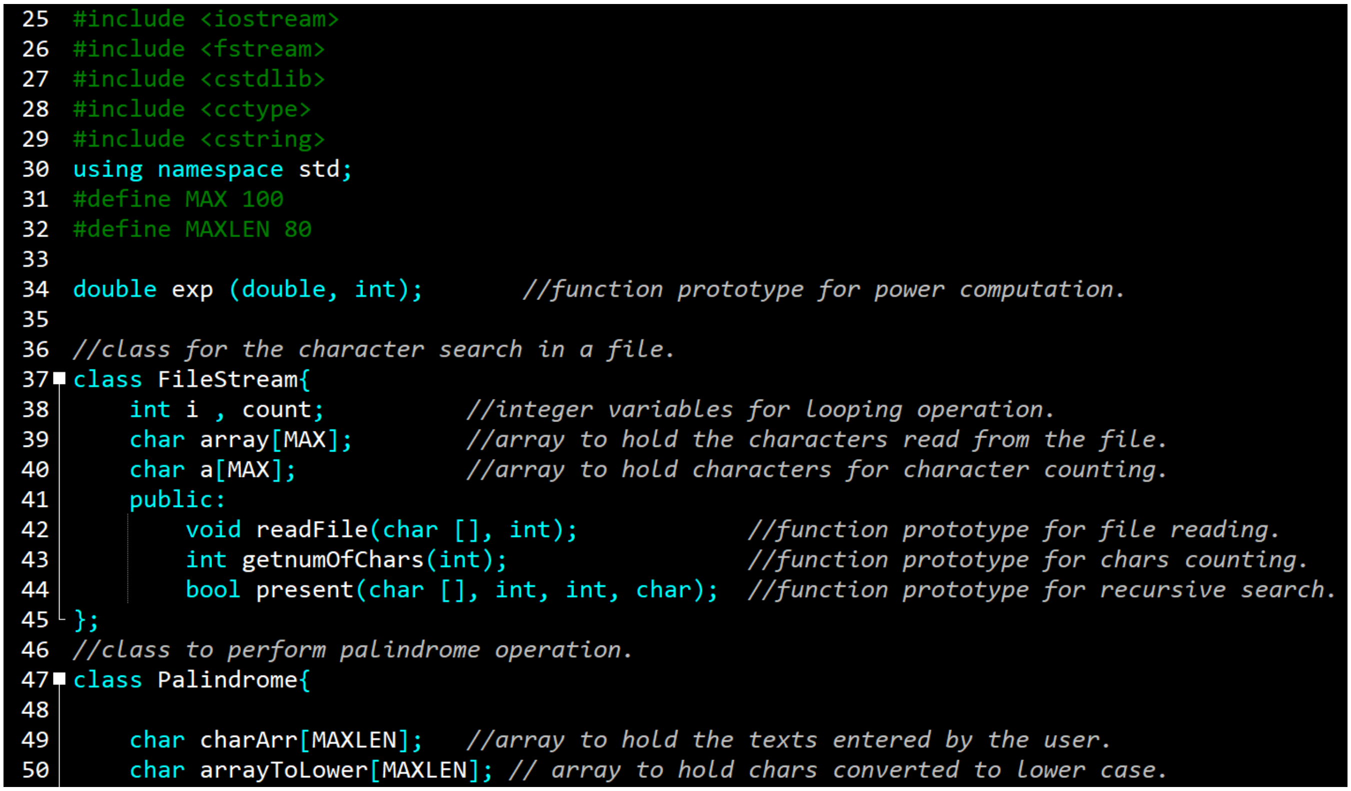
The main program initializes the `class Palindrome` and class `FileStream` and calls their respective member functions as necessary. The main function uses a `switch ()` statement inside a forever `while ()` loop to allow the user to perform different computations. There is also an exit condition if the user wants to quit running the program. While it calls and initializes the individual functions, the `main` will also display a series of messages to aware the user about what is going on.

Taddese Erba
Section – I
Project – Two
Due: January 30, 2020

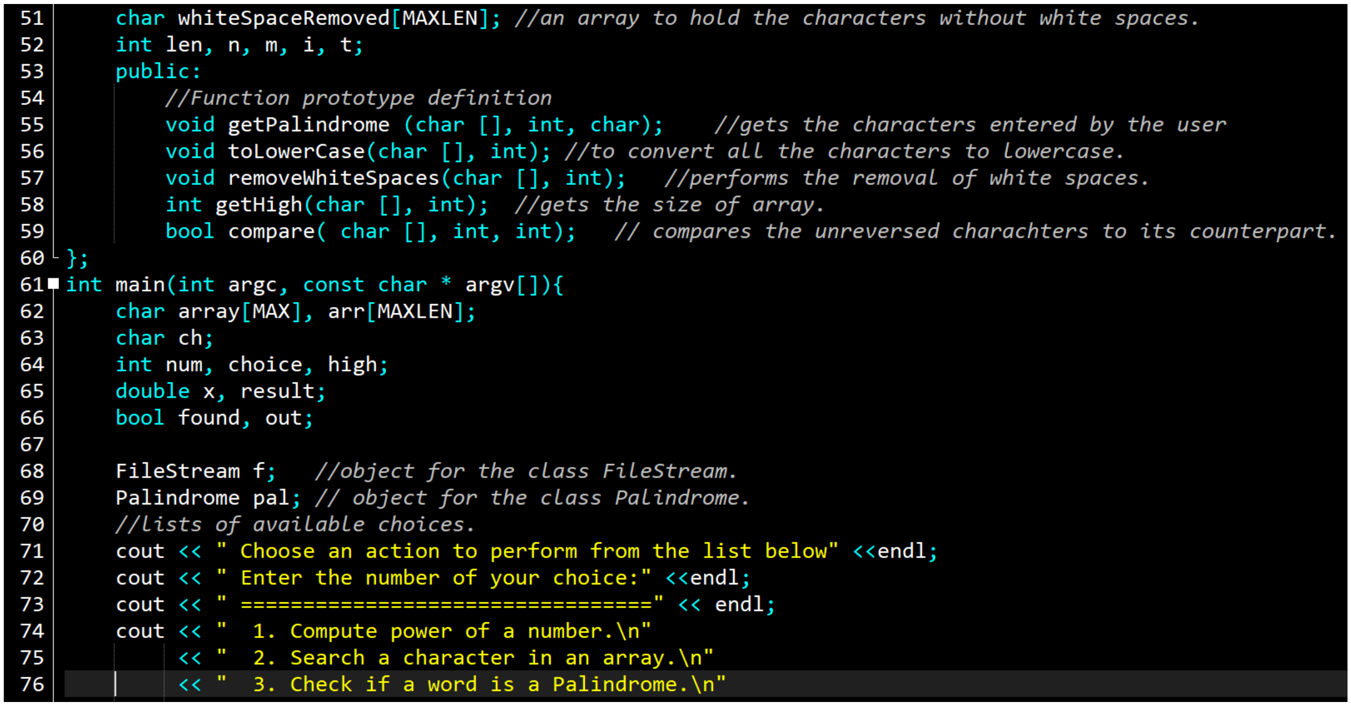
Code listing



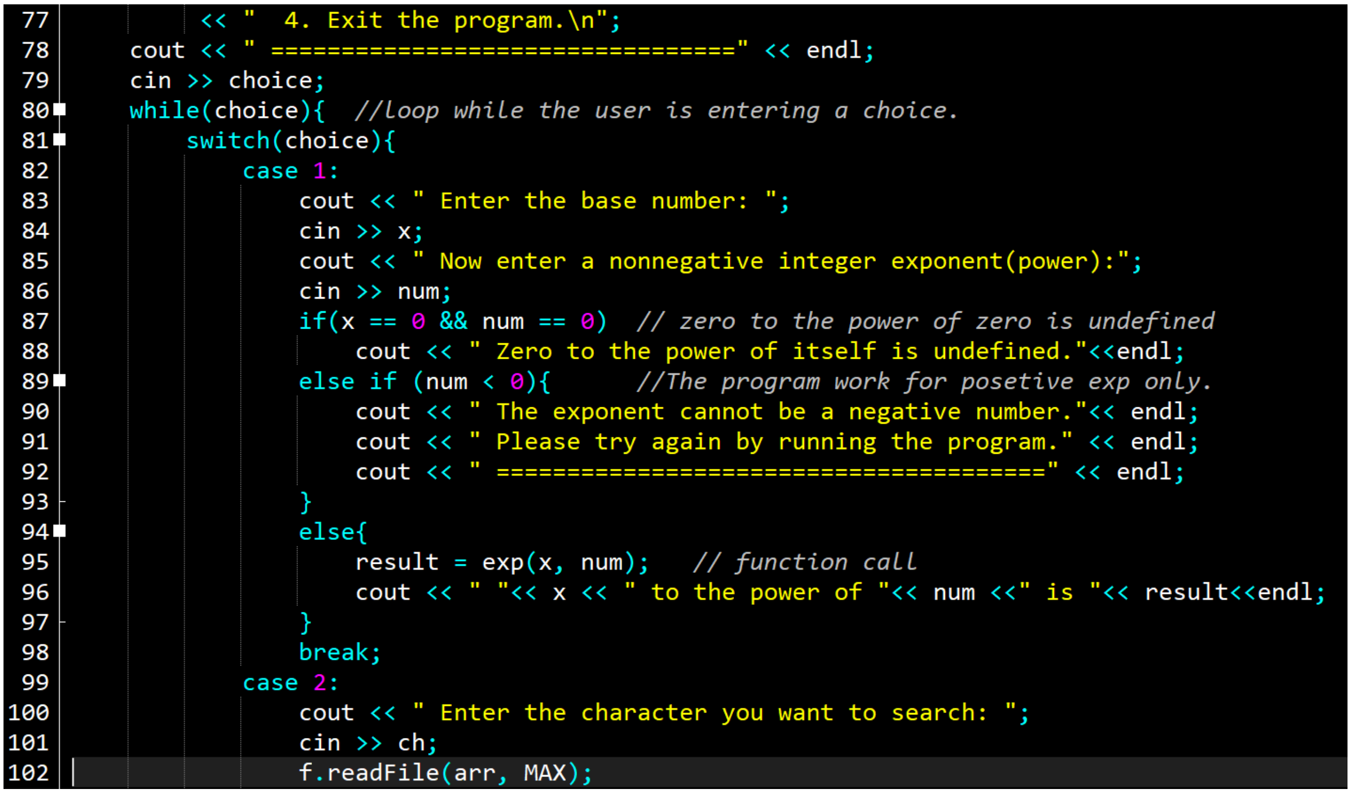
```
/*
This Program can perform the following tasks.
 1. It can calculate the power of any number raised to positive integers.
    *** To chieve this the program requests the user to enter the base
        number followed by another request to enter a non-negative intiger.
        if the user enters a negative integer, the program anounces it and exit.
        the program also consider 0 to the power of zero is undefined. ***
 2. It can receive a character from the user, serach a file for the presence of
    an alphanumeric character, and display a message
    whether the character is found in the file or not. The program will not
    search and return the right output if the user enter a string.
    *** If the user enter a string, the program may run or crush but will not work
        correctly. At times, it may search the array for the first character in the
        string and return the result. ***
 3. It can accept a word or phrase with number of characters not exceeding 80
    convert the all the characters to lower case, remove all the whitespaces
    from the words entered, and tell whether the word(s) is a palindrome or not.
    *** If the user enters a non-alphabetic characters, the program may output
        an unexpected result. The program is designed to work correctly with
        alphabetic characters. ***
    NOTE: to make running the program convinient for the user, the program uses infinite
          loop with switch() statment. The user can choose what to perform next. If the user
          wishes to exit, an exit condition is specified.
*/
```



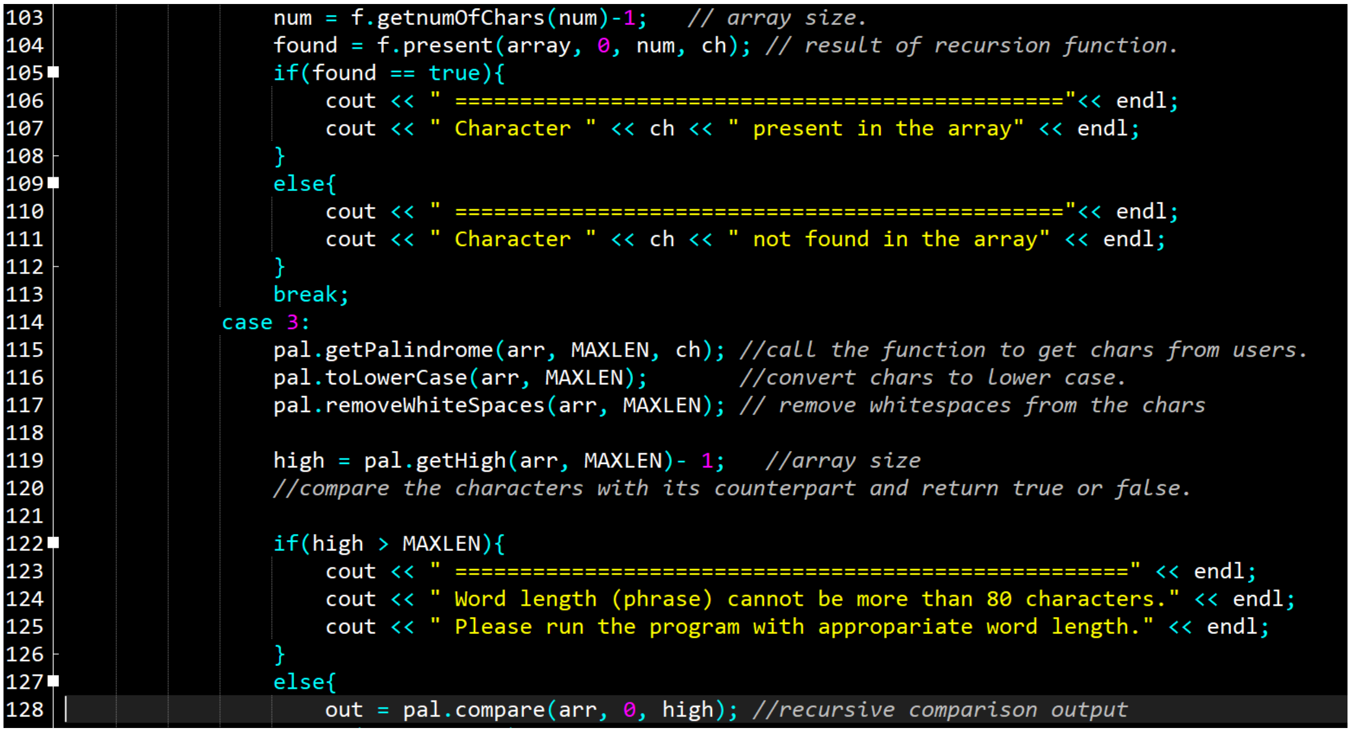
```
#include <iostream>
#include <fstream>
#include <cstdlib>
#include <cctype>
#include <cstring>
using namespace std;
#define MAX 100
#define MAXLEN 80

double exp (double, int);       //function prototype for power computation.

//class for the character search in a file.
class FileStream{
    int i , count;          //integer variables for looping operation.
    char array[MAX];        //array to hold the characters read from the file.
    char a[MAX];            //array to hold characters for character counting.
    public:
        void readFile(char [], int);            //function prototype for file reading.
        int getnumOfChars(int);                 //function prototype for chars counting.
        bool present(char [], int, int, char);  //function prototype for recursive search.
};
//class to perform palindrome operation.
class Palindrome{

    char charArr[MAXLEN];   //array to hold the texts entered by the user.
    char arrayToLower[MAXLEN]; // array to hold chars converted to lower case.
```



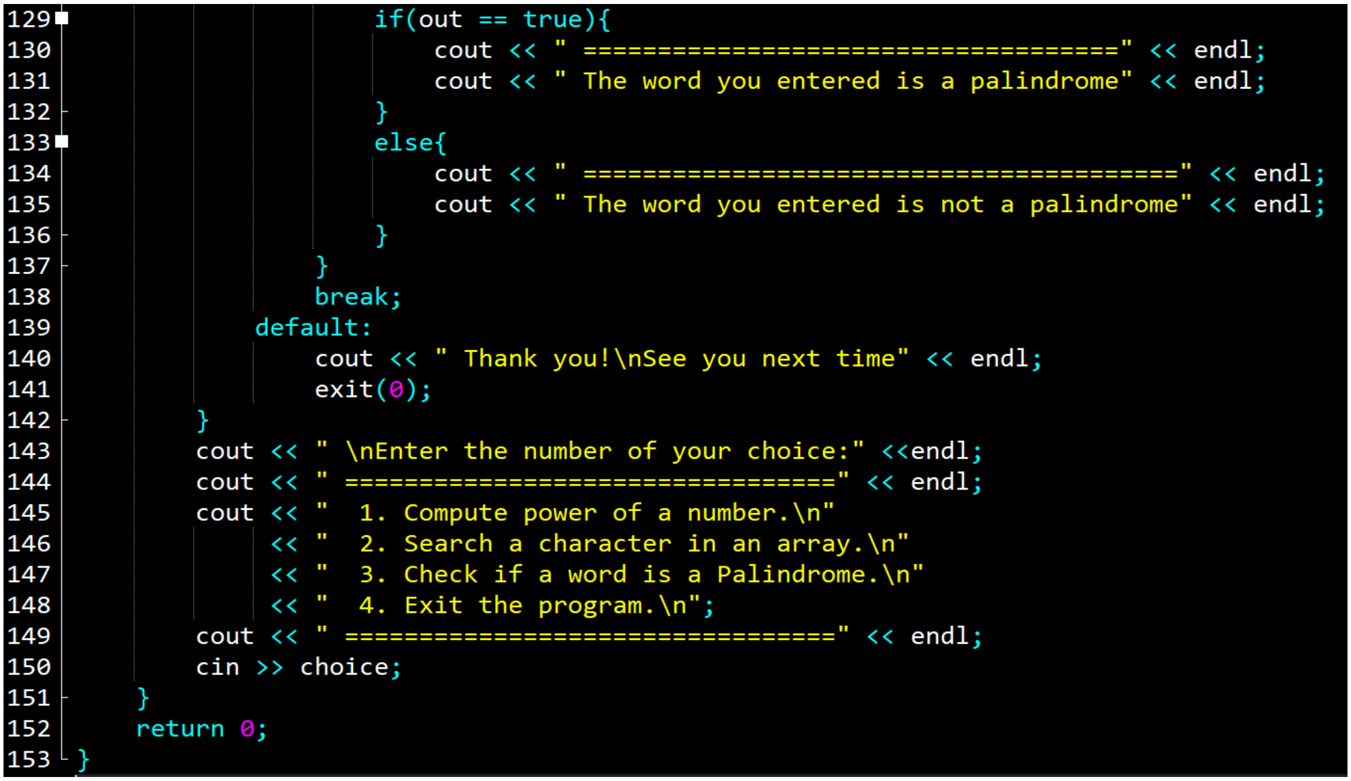
```
    char whiteSpaceRemoved[MAXLEN]; //an array to hold the characters without white spaces.
    int len, n, m, i, t;
    public:
        //Function prototype definition
        void getPalindrome (char [], int, char);    //gets the characters entered by the user
        void toLowerCase(char [], int); //to convert all the characters to lowercase.
        void removeWhiteSpaces(char [], int);   //performs the removal of white spaces.
        int getHigh(char [], int);  //gets the size of array.
        bool compare( char [], int, int);   // compares the unreversed charachters to its counterpart.
};
int main(int argc, const char * argv[]){
    char array[MAX], arr[MAXLEN];
    char ch;
    int num, choice, high;
    double x, result;
    bool found, out;

    FileStream f;   //object for the class FileStream.
    Palindrome pal; // object for the class Palindrome.
    //lists of available choices.
    cout << " Choose an action to perform from the list below" <<endl;
    cout << " Enter the number of your choice:" <<endl;
    cout << " =================================" << endl;
    cout << "  1. Compute power of a number.\n"
         << "  2. Search a character in an array.\n"
         << "  3. Check if a word is a Palindrome.\n"
```



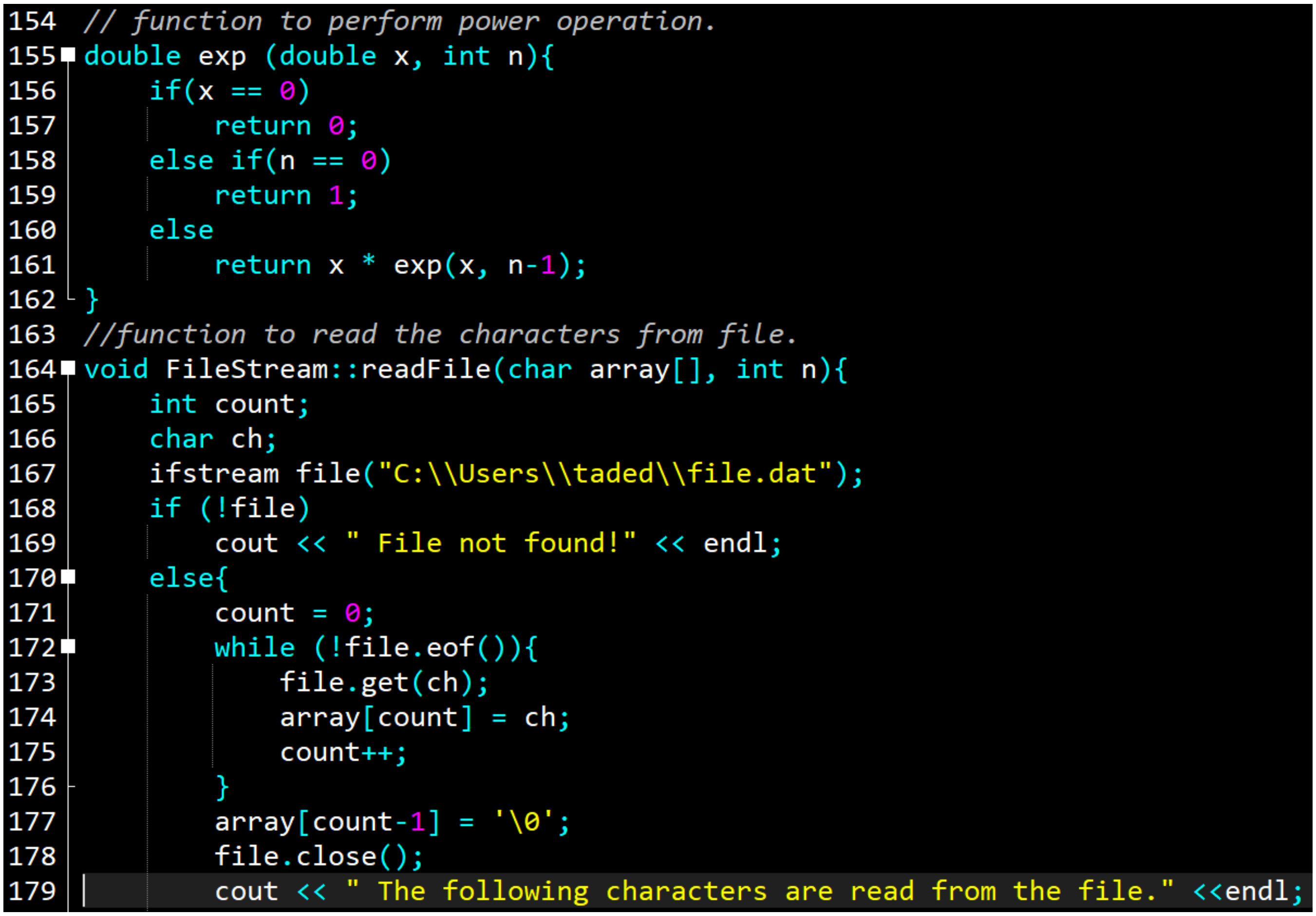
```
         << "  4. Exit the program.\n";
    cout << " =================================" << endl;
    cin >> choice;
    while(choice){  //loop while the user is entering a choice.
        switch(choice){
            case 1:
                cout << " Enter the base number: ";
                cin >> x;
                cout << " Now enter a nonnegative integer exponent(power):";
                cin >> num;
                if(x == 0 && num == 0)  // zero to the power of zero is undefined
                    cout << " Zero to the power of itself is undefined."<<endl;
                else if (num < 0){      //The program work for posetive exp only.
                    cout << " The exponent cannot be a negative number."<< endl;
                    cout << " Please try again by running the program." << endl;
                    cout << " =======================================" << endl;
                }
                else{
                    result = exp(x, num);   // function call
                    cout << " "<< x << " to the power of "<< num <<" is "<< result<<endl;
                }
                break;
            case 2:
                cout << " Enter the character you want to search: ";
                cin >> ch;
                f.readFile(arr, MAX);
```



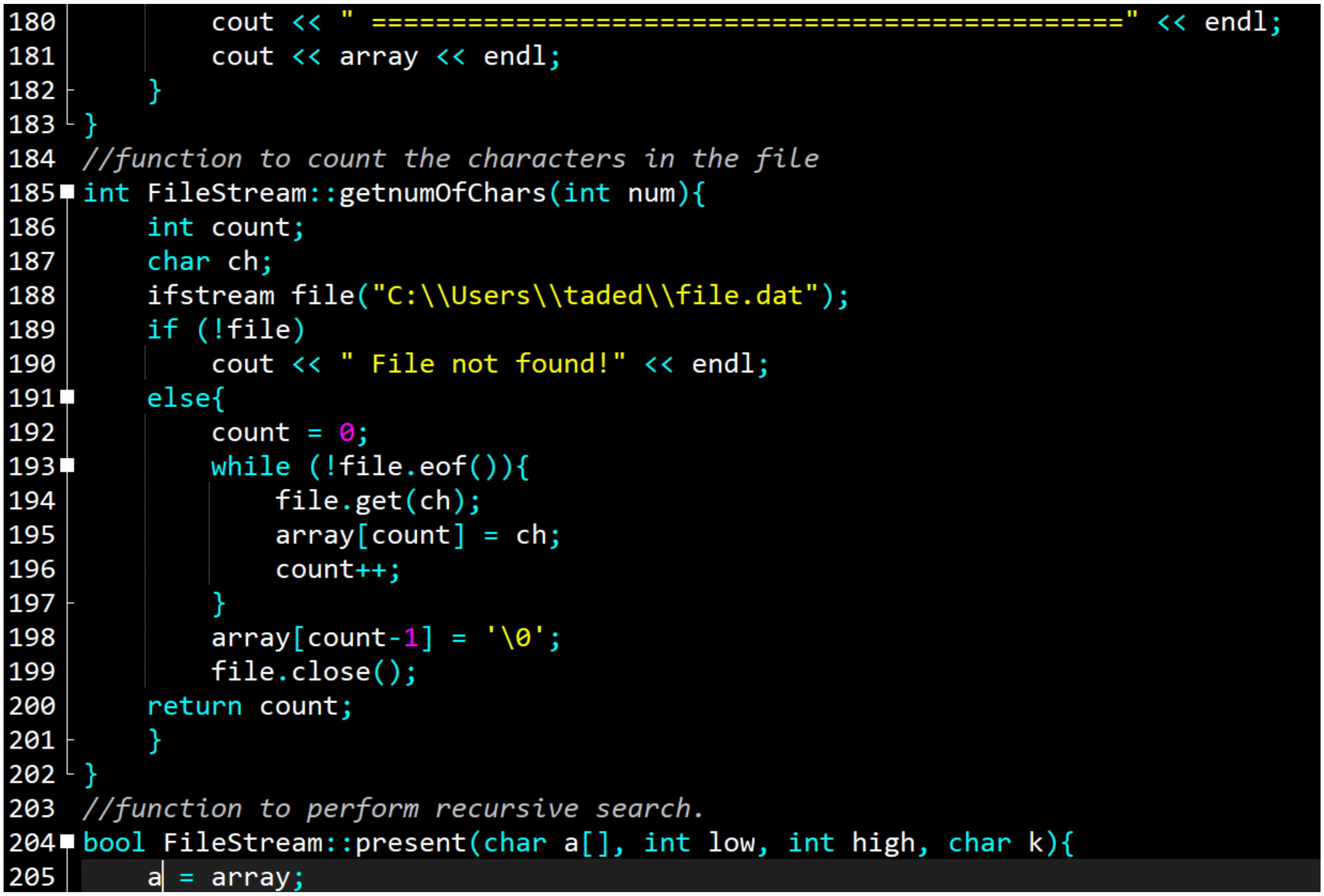
```cpp
                num = f.getnumOfChars(num)-1;   // array size.
                found = f.present(array, 0, num, ch); // result of recursion function.
                if(found == true){
                    cout << " ==============================================="<< endl;
                    cout << " Character " << ch << " present in the array" << endl;
                }
                else{
                    cout << " ==============================================="<< endl;
                    cout << " Character " << ch << " not found in the array" << endl;
                }
                break;
            case 3:
                pal.getPalindrome(arr, MAXLEN, ch); //call the function to get chars from users.
                pal.toLowerCase(arr, MAXLEN);       //convert chars to lower case.
                pal.removeWhiteSpaces(arr, MAXLEN); // remove whitespaces from the chars

                high = pal.getHigh(arr, MAXLEN)- 1;   //array size
                //compare the characters with its counterpart and return true or false.

                if(high > MAXLEN){
                    cout << " ====================================================" << endl;
                    cout << " Word length (phrase) cannot be more than 80 characters." << endl;
                    cout << " Please run the program with appropariate word length." << endl;
                }
                else{
                    out = pal.compare(arr, 0, high); //recursive comparison output
```



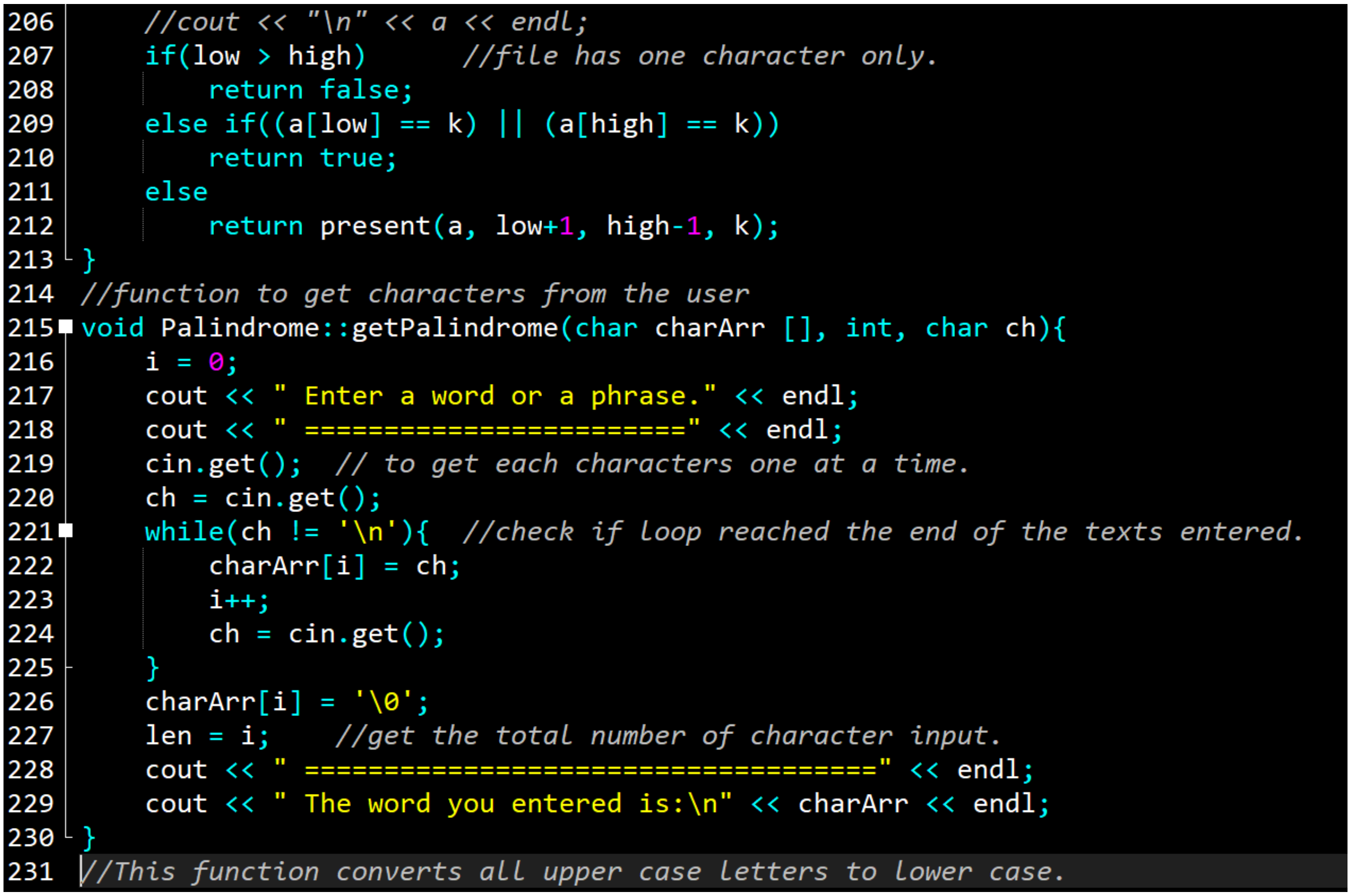
```cpp
                    if(out == true){
                        cout << " ====================================" << endl;
                        cout << " The word you entered is a palindrome" << endl;
                    }
                    else{
                        cout << " ========================================" << endl;
                        cout << " The word you entered is not a palindrome" << endl;
                    }
                }
                break;
            default:
                cout << " Thank you!\nSee you next time" << endl;
                exit(0);
        }
        cout << " \nEnter the number of your choice:" <<endl;
        cout << " =================================" << endl;
        cout << "  1. Compute power of a number.\n"
             << "  2. Search a character in an array.\n"
             << "  3. Check if a word is a Palindrome.\n"
             << "  4. Exit the program.\n";
        cout << " =================================" << endl;
        cin >> choice;
    }
    return 0;
}
```



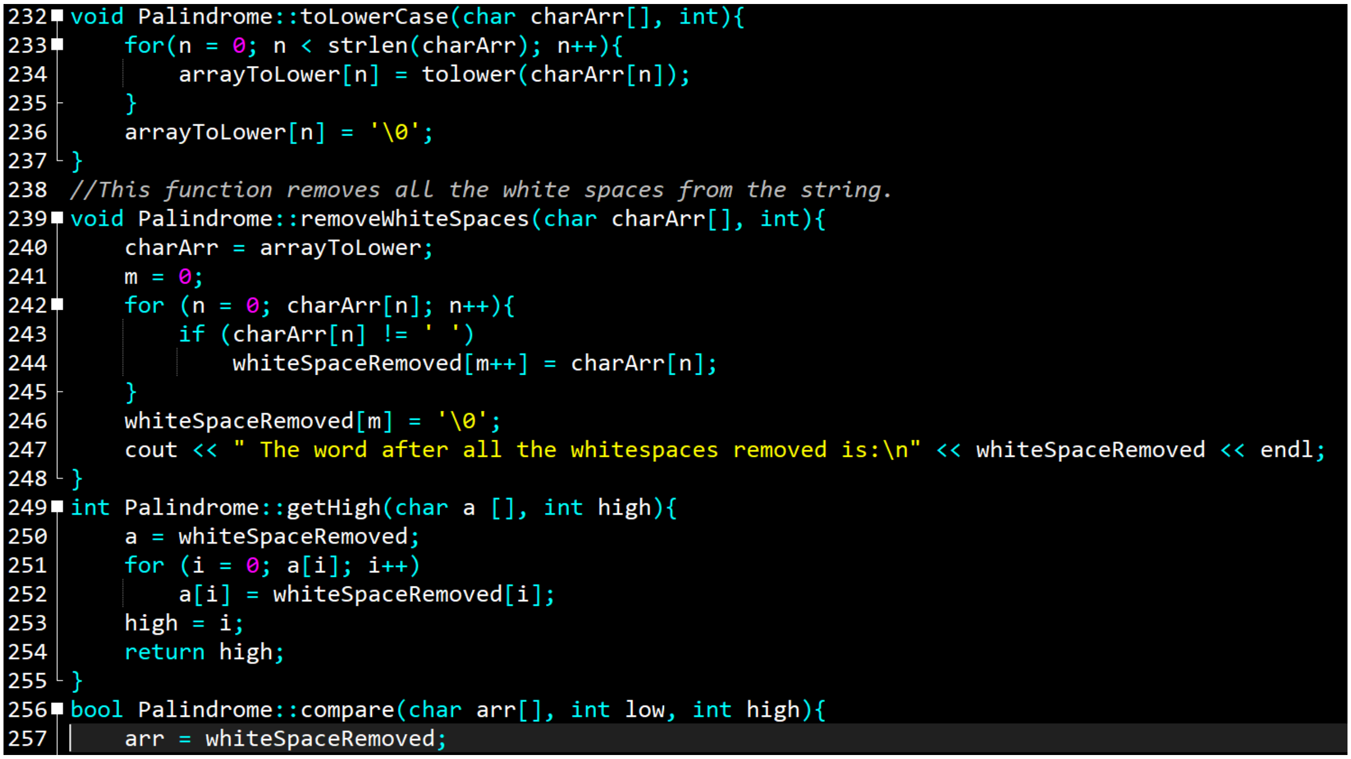
```
// function to perform power operation.
double exp (double x, int n){
    if(x == 0)
        return 0;
    else if(n == 0)
        return 1;
    else
        return x * exp(x, n-1);
}
//function to read the characters from file.
void FileStream::readFile(char array[], int n){
    int count;
    char ch;
    ifstream file("C:\\Users\\taded\\file.dat");
    if (!file)
        cout << " File not found!" << endl;
    else{
        count = 0;
        while (!file.eof()){
            file.get(ch);
            array[count] = ch;
            count++;
        }
        array[count-1] = '\0';
        file.close();
        cout << " The following characters are read from the file." <<endl;
```



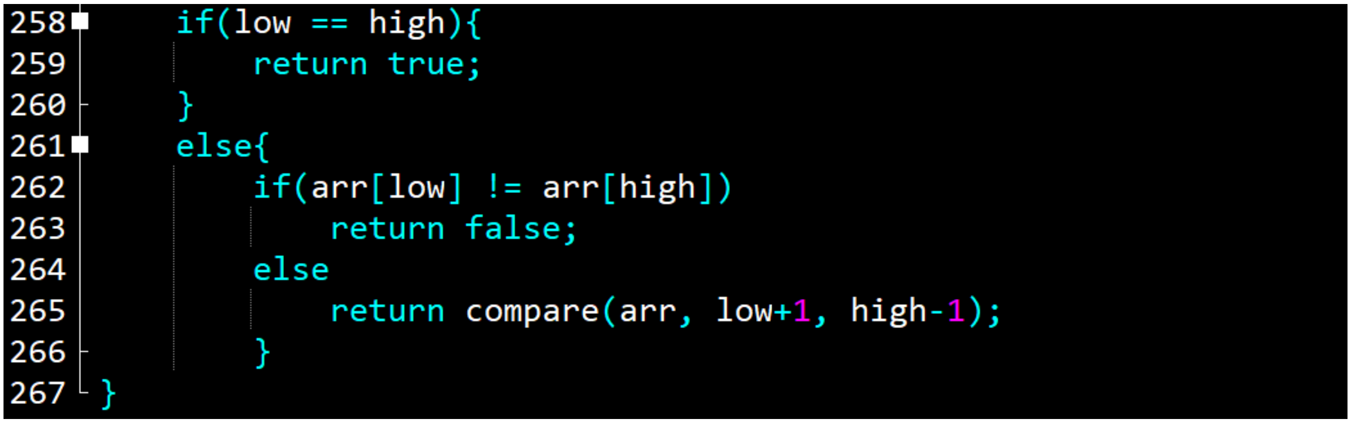
```
        cout << " ===============================================" << endl;
        cout << array << endl;
    }
}
//function to count the characters in the file
int FileStream::getnumOfChars(int num){
    int count;
    char ch;
    ifstream file("C:\\Users\\taded\\file.dat");
    if (!file)
        cout << " File not found!" << endl;
    else{
        count = 0;
        while (!file.eof()){
            file.get(ch);
            array[count] = ch;
            count++;
        }
        array[count-1] = '\0';
        file.close();
    return count;
    }
}
//function to perform recursive search.
bool FileStream::present(char a[], int low, int high, char k){
    a = array;
```



```cpp
    //cout << "\n" << a << endl;
    if(low > high)      //file has one character only.
        return false;
    else if((a[low] == k) || (a[high] == k))
        return true;
    else
        return present(a, low+1, high-1, k);
}
//function to get characters from the user
void Palindrome::getPalindrome(char charArr [], int, char ch){
    i = 0;
    cout << " Enter a word or a phrase." << endl;
    cout << " ========================" << endl;
    cin.get();  // to get each characters one at a time.
    ch = cin.get();
    while(ch != '\n'){  //check if loop reached the end of the texts entered.
        charArr[i] = ch;
        i++;
        ch = cin.get();
    }
    charArr[i] = '\0';
    len = i;    //get the total number of character input.
    cout << " ===================================" << endl;
    cout << " The word you entered is:\n" << charArr << endl;
}
//This function converts all upper case letters to lower case.
```



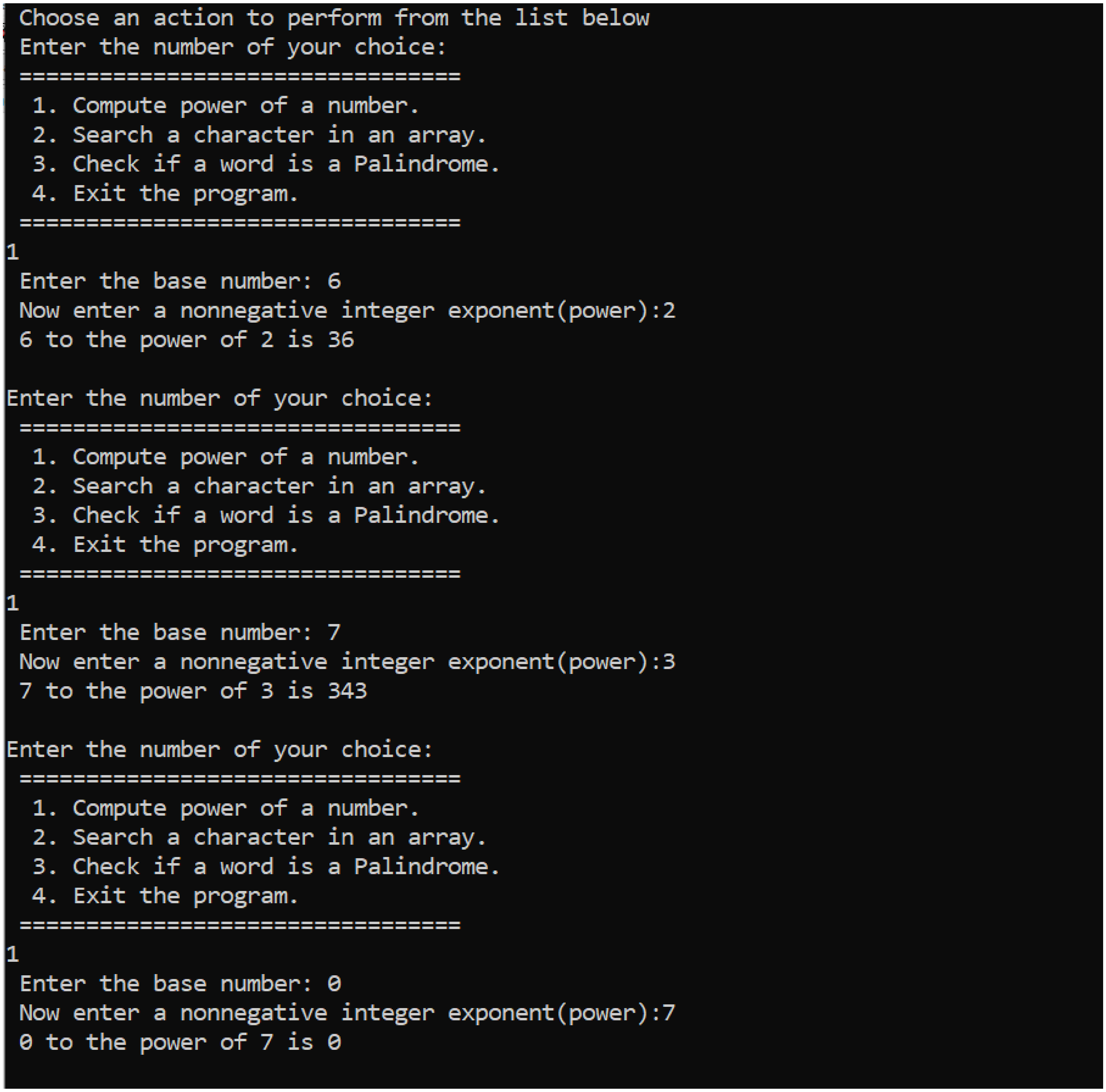
```
void Palindrome::toLowerCase(char charArr[], int){
    for(n = 0; n < strlen(charArr); n++){
        arrayToLower[n] = tolower(charArr[n]);
    }
    arrayToLower[n] = '\0';
}
//This function removes all the white spaces from the string.
void Palindrome::removeWhiteSpaces(char charArr[], int){
    charArr = arrayToLower;
    m = 0;
    for (n = 0; charArr[n]; n++){
        if (charArr[n] != ' ')
            whiteSpaceRemoved[m++] = charArr[n];
    }
    whiteSpaceRemoved[m] = '\0';
    cout << " The word after all the whitespaces removed is:\n" << whiteSpaceRemoved << endl;
}
int Palindrome::getHigh(char a [], int high){
    a = whiteSpaceRemoved;
    for (i = 0; a[i]; i++)
        a[i] = whiteSpaceRemoved[i];
    high = i;
    return high;
}
bool Palindrome::compare(char arr[], int low, int high){
    arr = whiteSpaceRemoved;
```



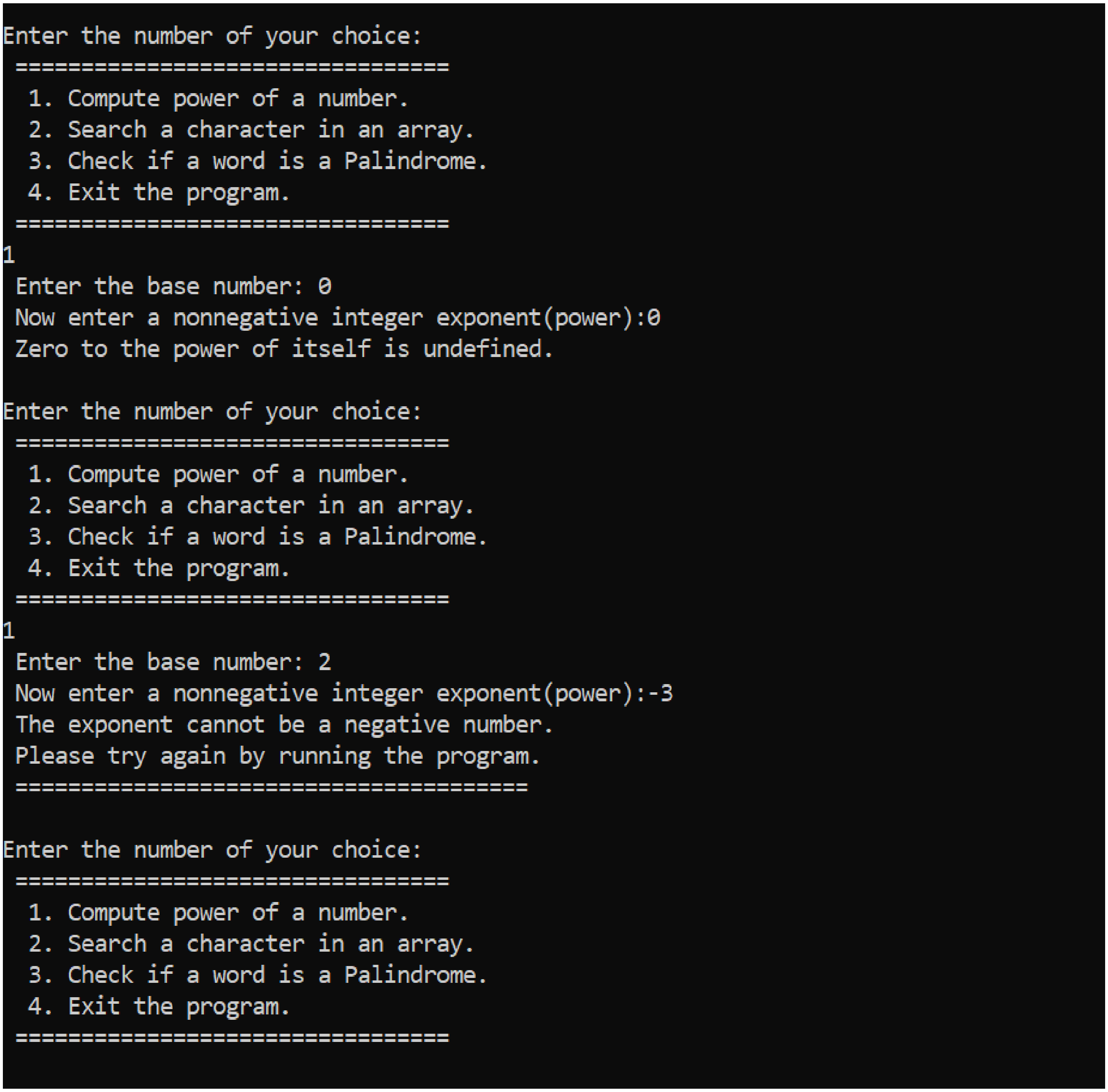
```
    if(low == high){
        return true;
    }
    else{
        if(arr[low] != arr[high])
            return false;
        else
            return compare(arr, low+1, high-1);
        }
}
```

Test Results

1. Tests result for exponential computation.

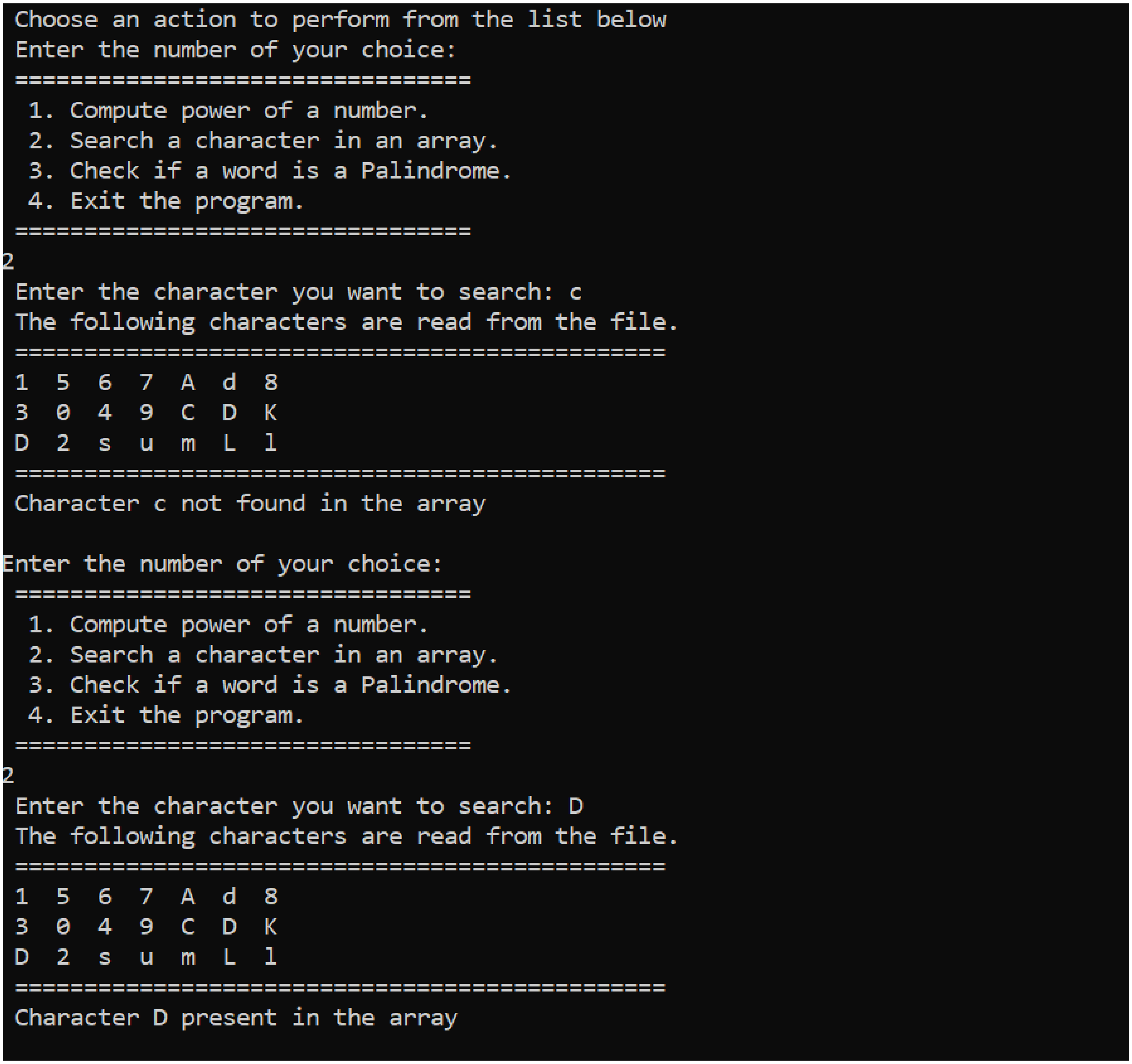


```
 Choose an action to perform from the list below
 Enter the number of your choice:
 ==================================
  1. Compute power of a number.
  2. Search a character in an array.
  3. Check if a word is a Palindrome.
  4. Exit the program.
 ==================================
1
 Enter the base number: 6
 Now enter a nonnegative integer exponent(power):2
 6 to the power of 2 is 36

Enter the number of your choice:
 ==================================
  1. Compute power of a number.
  2. Search a character in an array.
  3. Check if a word is a Palindrome.
  4. Exit the program.
 ==================================
1
 Enter the base number: 7
 Now enter a nonnegative integer exponent(power):3
 7 to the power of 3 is 343

Enter the number of your choice:
 ==================================
  1. Compute power of a number.
  2. Search a character in an array.
  3. Check if a word is a Palindrome.
  4. Exit the program.
 ==================================
1
 Enter the base number: 0
 Now enter a nonnegative integer exponent(power):7
 0 to the power of 7 is 0
```



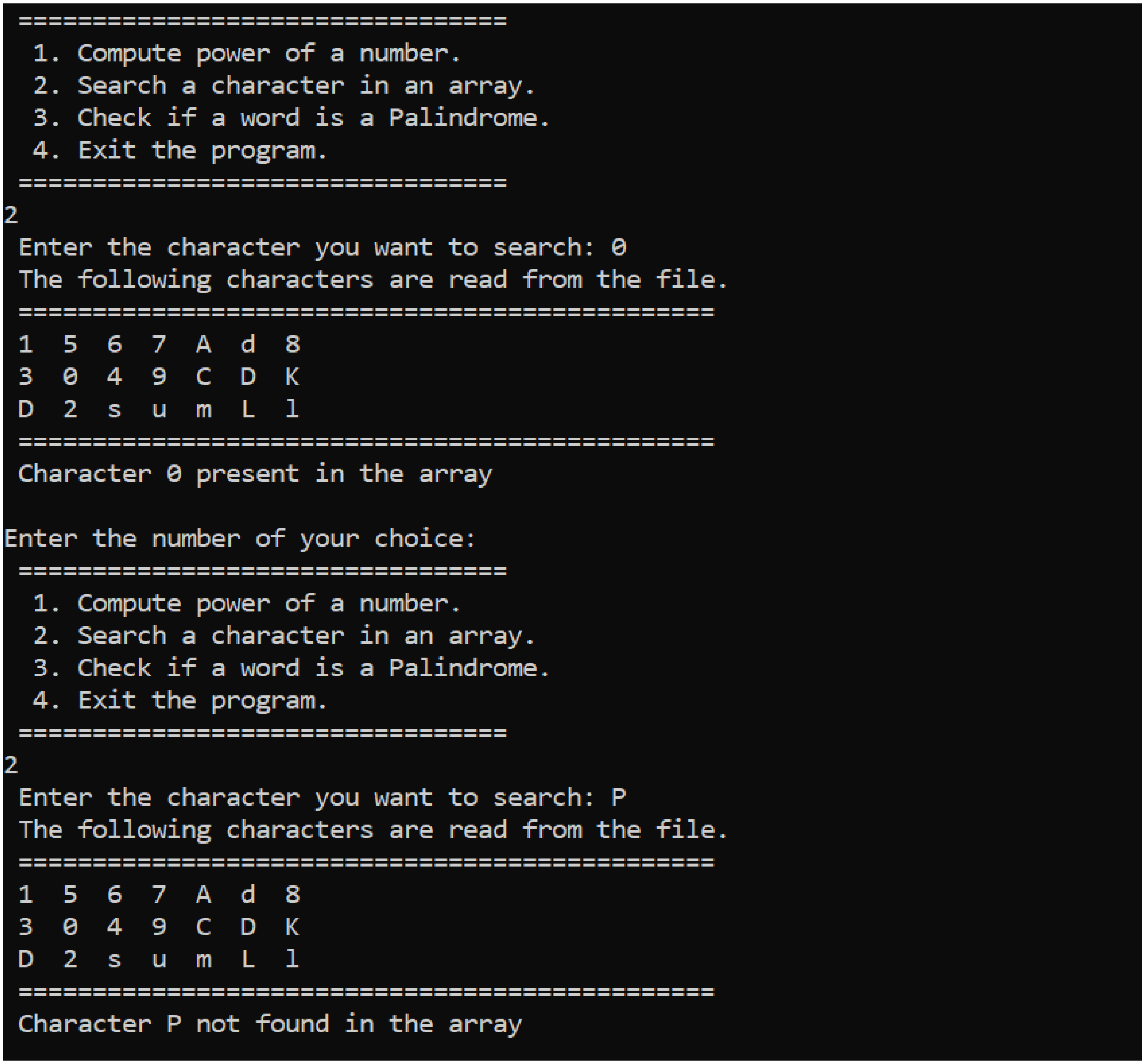
```
Enter the number of your choice:
 =================================
  1. Compute power of a number.
  2. Search a character in an array.
  3. Check if a word is a Palindrome.
  4. Exit the program.
 =================================
1
 Enter the base number: 0
 Now enter a nonnegative integer exponent(power):0
 Zero to the power of itself is undefined.

Enter the number of your choice:
 =================================
  1. Compute power of a number.
  2. Search a character in an array.
  3. Check if a word is a Palindrome.
  4. Exit the program.
 =================================
1
 Enter the base number: 2
 Now enter a nonnegative integer exponent(power):-3
 The exponent cannot be a negative number.
 Please try again by running the program.
 =======================================

Enter the number of your choice:
 =================================
  1. Compute power of a number.
  2. Search a character in an array.
  3. Check if a word is a Palindrome.
  4. Exit the program.
 =================================
```

2. Tests result for character search in a file.



```
 Choose an action to perform from the list below
 Enter the number of your choice:
 =================================
  1. Compute power of a number.
  2. Search a character in an array.
  3. Check if a word is a Palindrome.
  4. Exit the program.
 =================================
2
 Enter the character you want to search: c
 The following characters are read from the file.
 ===============================================
 1  5  6  7  A  d  8
 3  0  4  9  C  D  K
 D  2  s  u  m  L  l
 ===============================================
 Character c not found in the array

Enter the number of your choice:
 =================================
  1. Compute power of a number.
  2. Search a character in an array.
  3. Check if a word is a Palindrome.
  4. Exit the program.
 =================================
2
 Enter the character you want to search: D
 The following characters are read from the file.
 ===============================================
 1  5  6  7  A  d  8
 3  0  4  9  C  D  K
 D  2  s  u  m  L  l
 ===============================================
 Character D present in the array
```

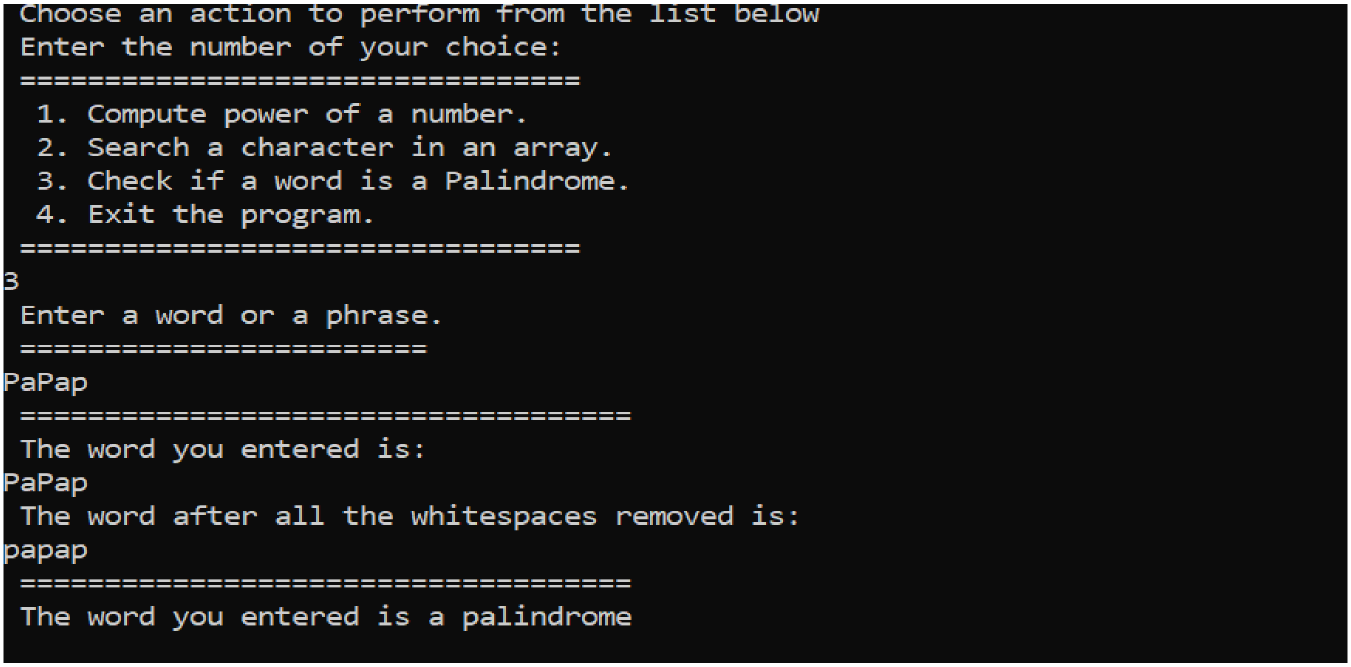
Taddese Erba
Section – I
Project – Two
Due: January 30, 2020



```
 =================================
  1. Compute power of a number.
  2. Search a character in an array.
  3. Check if a word is a Palindrome.
  4. Exit the program.
 =================================
2
 Enter the character you want to search: 0
 The following characters are read from the file.
 ===============================================
 1  5  6  7  A  d  8
 3  0  4  9  C  D  K
 D  2  s  u  m  L  l
 ===============================================
 Character 0 present in the array

Enter the number of your choice:
 =================================
  1. Compute power of a number.
  2. Search a character in an array.
  3. Check if a word is a Palindrome.
  4. Exit the program.
 =================================
2
 Enter the character you want to search: P
 The following characters are read from the file.
 ===============================================
 1  5  6  7  A  d  8
 3  0  4  9  C  D  K
 D  2  s  u  m  L  l
 ===============================================
 Character P not found in the array
```

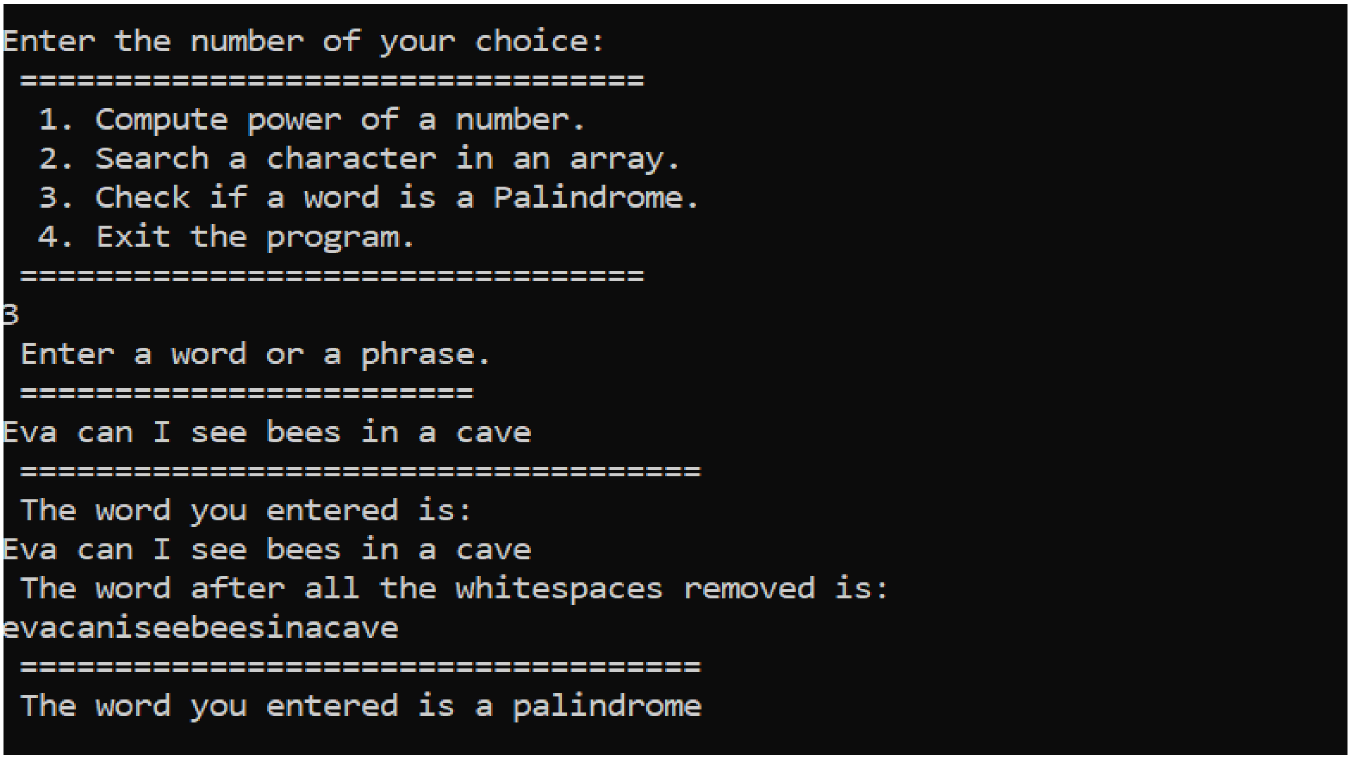
3. Tests with a palindrome word

a. Short (single) word



```
 Choose an action to perform from the list below
 Enter the number of your choice:
 =================================
  1. Compute power of a number.
  2. Search a character in an array.
  3. Check if a word is a Palindrome.
  4. Exit the program.
 =================================
3
 Enter a word or a phrase.
 ========================
PaPap
 ====================================
 The word you entered is:
PaPap
 The word after all the whitespaces removed is:
papap
 ====================================
 The word you entered is a palindrome
```

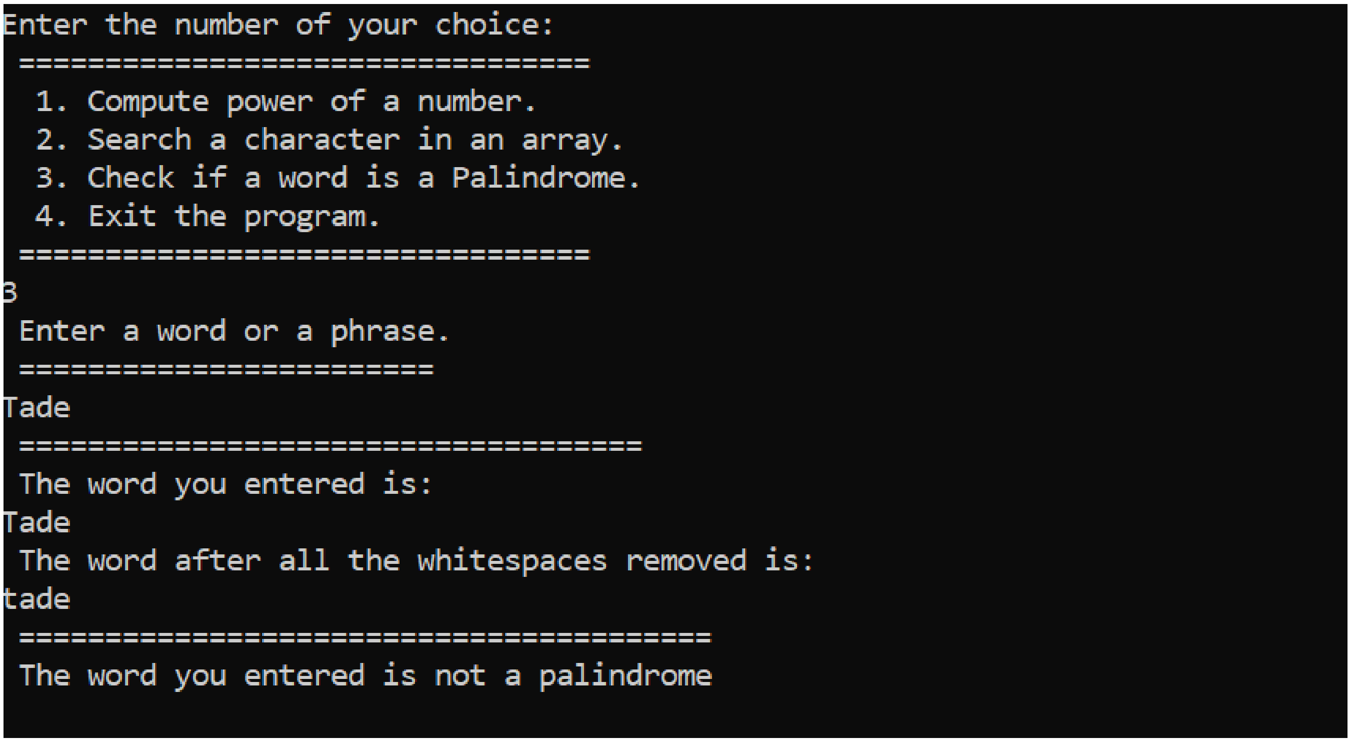
b. Long word



```
Enter the number of your choice:
 =================================
  1. Compute power of a number.
  2. Search a character in an array.
  3. Check if a word is a Palindrome.
  4. Exit the program.
 =================================
3
 Enter a word or a phrase.
 ========================
Eva can I see bees in a cave
 ====================================
 The word you entered is:
Eva can I see bees in a cave
 The word after all the whitespaces removed is:
evacaniseebeesinacave
 ====================================
 The word you entered is a palindrome
```

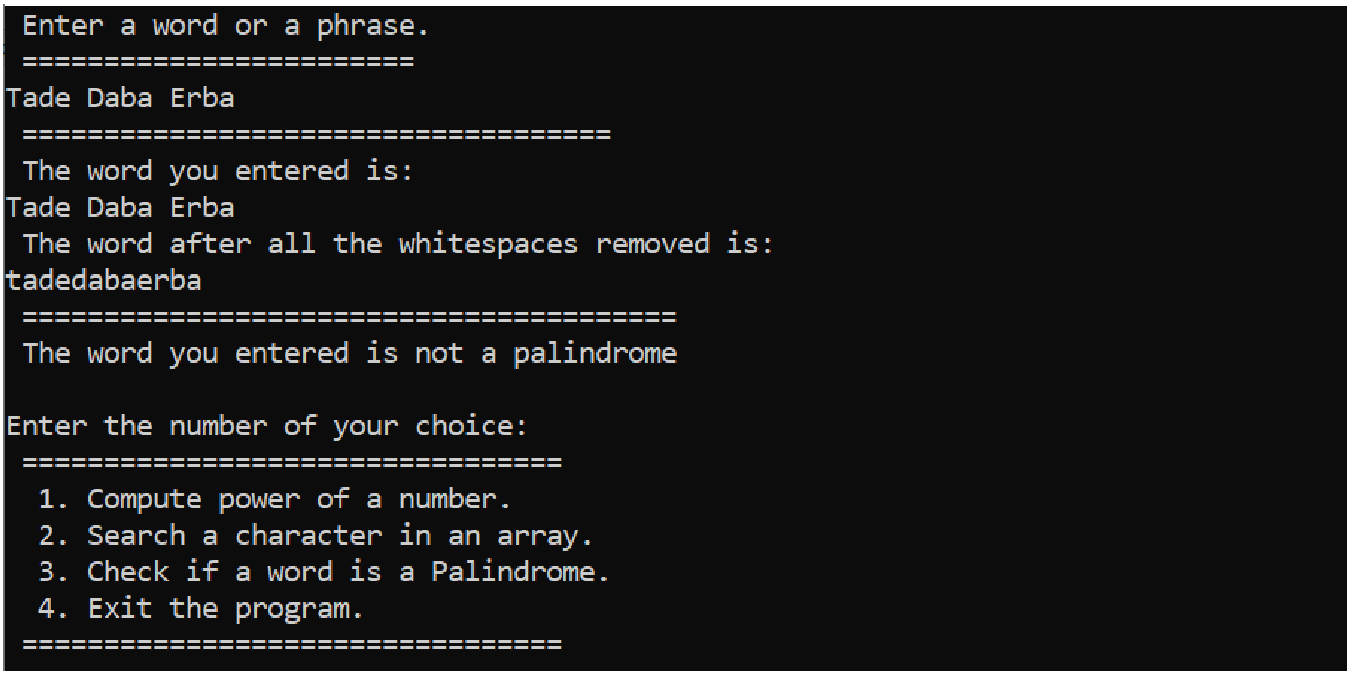
3.1 Tests with a nonpalindromic word

a. Short (single) word



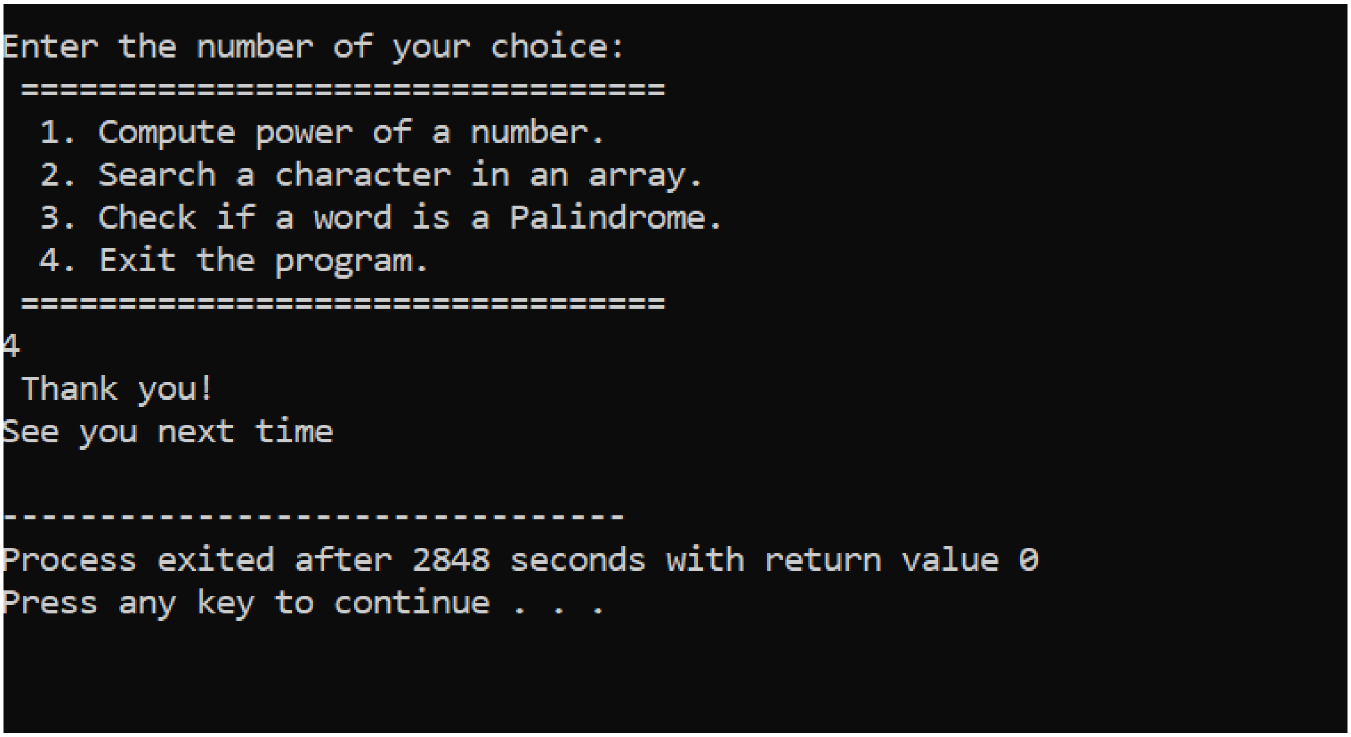
```
Enter the number of your choice:
 =================================
  1. Compute power of a number.
  2. Search a character in an array.
  3. Check if a word is a Palindrome.
  4. Exit the program.
 =================================
3
 Enter a word or a phrase.
 ========================
Tade
 ====================================
 The word you entered is:
Tade
 The word after all the whitespaces removed is:
tade
 ========================================
 The word you entered is not a palindrome
```

b. Long word



```
 Enter a word or a phrase.
 ========================
Tade Daba Erba
 ====================================
 The word you entered is:
Tade Daba Erba
 The word after all the whitespaces removed is:
tadedabaerba
 ========================================
 The word you entered is not a palindrome

Enter the number of your choice:
 =================================
  1. Compute power of a number.
  2. Search a character in an array.
  3. Check if a word is a Palindrome.
  4. Exit the program.
 =================================
```

4. Tests for exit condition.



```
Enter the number of your choice:
 =================================
  1. Compute power of a number.
  2. Search a character in an array.
  3. Check if a word is a Palindrome.
  4. Exit the program.
 =================================
4
 Thank you!
See you next time

--------------------------------
Process exited after 2848 seconds with return value 0
Press any key to continue . . .
```

User document

This program consists of three subprograms that perform different tasks. You can do the following by using the program.

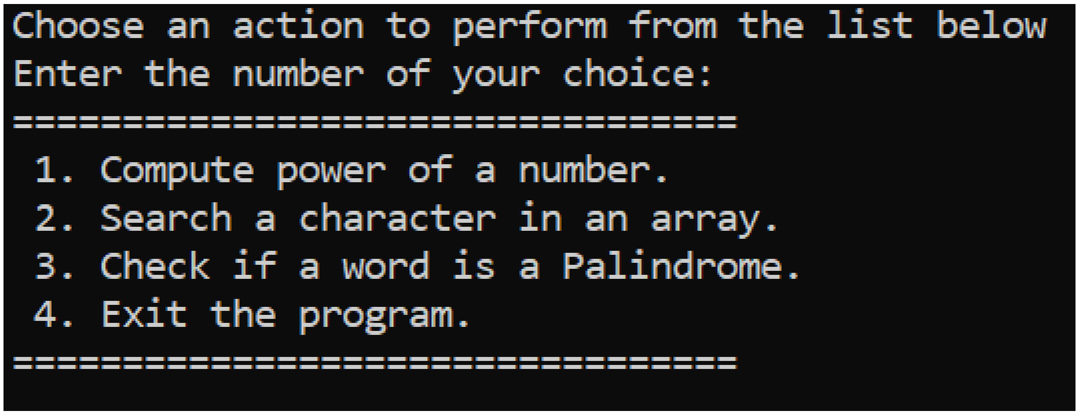
1. Compute the power of any number to a positive integer number.
2. You can search for a character in a file saved on your hard drive.
3. You can check if a word is a palindrome or not by entering a word from the terminal.

In order to run the program, you must perform the following steps.

☞ The program name is `Exponential.cpp.` on the terminal enter the following command to compile and run the program.

```
g++ -o Exponential Exponential.cpp
```

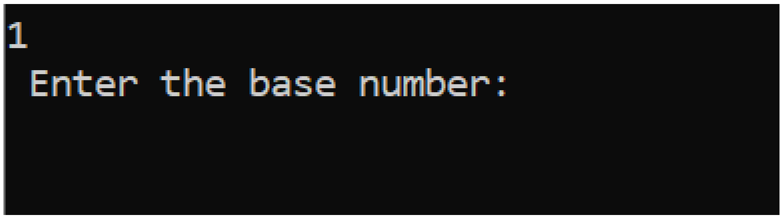
☞ The program will compile and open the following:



```
Choose an action to perform from the list below
Enter the number of your choice:
=================================
 1. Compute power of a number.
 2. Search a character in an array.
 3. Check if a word is a Palindrome.
 4. Exit the program.
=================================
```

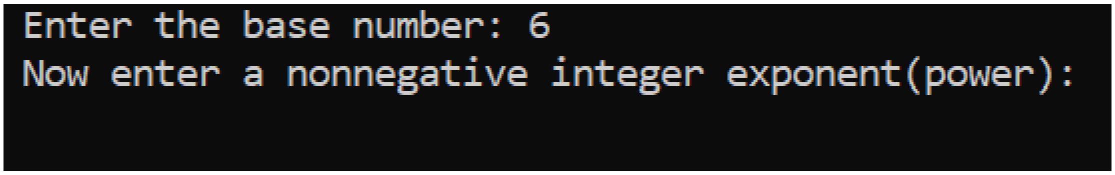
☞ Make a choice from the displayed menu. For example to compute exponential, type 1 and hit enter.

Then the program will ask you to enter a number as shown below.



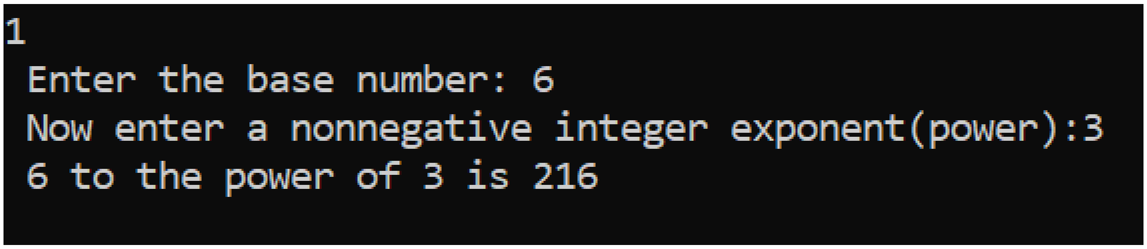
```
1
 Enter the base number:
```

☞ Enter a number. The program will display the following.



```
 Enter the base number: 6
 Now enter a nonnegative integer exponent(power):
```

☞ Now enter only a positive integer. The program doesn't work with any other number.



```
1
 Enter the base number: 6
 Now enter a nonnegative integer exponent(power):3
 6 to the power of 3 is 216
```

☞ Finally, the program will display the result as shown above.

The steps to perform the rest of the tasks is similar except you will require to type in the word you want to check for a palindrome and you also need to save the file you want to search as `file.dat` at the following location .

`C:\Users\taded\file.date`

Taddese Erba
Section – I
Project – Two
Due: January 30, 2020

Summery

This program is the collection of three stand-alone programs that can perform independently. The `double exp(int)` function exists to perform power computation. Its argument is an integer type indication it can take only an integer and can return a double or an integer. The classes `FileStream` and `Palindrome` are also stand-alone programs that can operate on character type arrays. The `main` function uses `while()` loop in conjunction with the `switch()` statement to create convenience for the user to perform the three independent tasks without the need to restart and run the program time and again.

This program can be used to perform mathematical computation (power computations), to implement search algorithms, and to control the size and nature of the characters entered by the user into some character limiter fields such as username and password fields. The program can be further improved by modifying the functions. For instance, the character search program can be improved to make a string search where the string size can be a character or the whole file or paragraph. Similarly, the palindrome program can be improved in a way it can perform interesting tasks such us limiting the character type and size entered by the user into some character accepting fields such as usernames and passwords. By completing this project, I have gained enough knowledge and experience to work with recursive functions and their implementation. One interesting thing about recursive functions that I came across as I complete this project is their simplicity in writing in programming code.