Text File Analysis Project

Program contains 2 different functions. They are:
main(): Function that runs the whole project.
letterFromPosition(int): takes integer representation of a letter and return that particular letter.

Code Operation

1) Prompts user for name of text file (including extension of the file, example: test.txt).
2) Allocates recommended memory to a variable to hold file content using "malloc".
3) Reads file into the variable (returns "Error: Could not find your file" if not able to read file).
4) Scan through file content and store each word into an array.
5) During scanning, programs counts:
   - **i.** The total number of words in file.
   - **ii.** The total number of characters in the file.
   - **iii.** Number of times each letter starts difference words in the file.
6) Programs checks letter that starts most words with the help of the **letterFromPosition** function.
7) Finally, program prints the statistics nicely.

Major Variables

1) **fileName(char):** holds file name taken from user.
2) **fileContent(char):** pointer to content read from text file.
3) **filePointer(FILE):** pointer to file.
4) **wordCount(int):** holds total number of words in file.
5) **characterCount(int):** holds total number of characters in file.
6) **mostWordStart(char):** holds letter that starts most words in file.
7) **wordStartCount(int):** holds the count of the letter that starts most words.
8) **WordStarterMonitor(int):** array of how many times each letter starts different words.

Source Code

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <ctype.h>
#define FILE_NAME_LENGTH 20

// function declarations
char letterFromPosition(int);

// main function
int main()
{
    // get text file name from user
    char fileName[FILE_NAME_LENGTH];
    printf("Enter the name of the text file. Example: test.txt: ");
    scanf("%s", &fileName);

    const char *fileNamePointer = fileName;
```

```
// generate recommended memory size for file reading
struct stat sb;
stat(fileNamePointer, &sb);
char *fileContent = malloc(sb.st_size);

// open file
FILE *filePointer;
if ((filePointer = fopen(fileNamePointer, "r")) == NULL)
{
    printf("Error: Could not find your file");
    // exit program
    exit(1);
}

// variables for analyzing text file
int wordCount = 0;
int characterCount = 0;
char mostWordStart;
int wordStartCount = 0;
int wordStarterMonitor[26] = {0};

// read text file word by word including punctuations
while (fscanf(filePointer, "%s", fileContent) != EOF)
{
    // increase number of words
    wordCount++;

    // count number of characters in word including punctuation
    for (int i = 0; fileContent[i] != '\0'; i++)
    {
        // increase number of characters
        characterCount++;
    }

    // check number of words that start with each letter (case insensitive)
    switch (toupper(fileContent[0]))
    {
    case 'A':
        wordStarterMonitor[0]++;
        break;
    case 'B':
        wordStarterMonitor[1]++;
        break;
    case 'C':
        wordStarterMonitor[2]++;
        break;
    case 'D':
        wordStarterMonitor[3]++;
    break;
case 'E':
    wordStarterMonitor[4]++;
    break;
case 'F':
    wordStarterMonitor[5]++;
    break;
case 'G':
    wordStarterMonitor[6]++;
    break;
case 'H':
    wordStarterMonitor[7]++;
    break;
case 'I':
    wordStarterMonitor[8]++;
    break;
case 'J':
    wordStarterMonitor[9]++;
    break;
case 'K':
    wordStarterMonitor[10]++;
    break;
case 'L':
    wordStarterMonitor[11]++;
    break;
case 'M':
    wordStarterMonitor[12]++;
    break;
case 'N':
    wordStarterMonitor[13]++;
    break;
case 'O':
    wordStarterMonitor[14]++;
    break;
case 'P':
    wordStarterMonitor[15]++;
    break;
case 'Q':
    wordStarterMonitor[16]++;
    break;
case 'R':
    wordStarterMonitor[17]++;
    break;
case 'S':
    wordStarterMonitor[18]++;
    break;
case 'T':
    wordStarterMonitor[19]++;
    break;
    case 'U':
        wordStarterMonitor[20]++;
        break;
    case 'V':
        wordStarterMonitor[21]++;
        break;
    case 'W':
        wordStarterMonitor[22]++;
        break;
    case 'X':
        wordStarterMonitor[23]++;
        break;
    case 'Y':
        wordStarterMonitor[24]++;
        break;
    case 'Z':
        wordStarterMonitor[25]++;
        break;
    default:
        break;
    }
}

// check letter that starts most words
for (int i = 0; i < 26; i++)
{
    if (wordStarterMonitor[i] > wordStartCount)
    {
        wordStartCount = wordStarterMonitor[i];
        mostWordStart = letterFromPosition(i);
    }
}

// print all statistics
printf("\n\nThe total number of words are: %d\n\n", wordCount);
printf("\n\nThe total number of characters are: %d\n\n", characterCount);

printf("\n\nThe number of words starting with A: %d\n", wordStarterMonitor[0]);
printf("\nThe number of words starting with B: %d\n", wordStarterMonitor[1]);
printf("\nThe number of words starting with C: %d\n", wordStarterMonitor[2]);
printf("\nThe number of words starting with D: %d\n", wordStarterMonitor[3]);
printf("\nThe number of words starting with E: %d\n", wordStarterMonitor[4]);
printf("\nThe number of words starting with F: %d\n", wordStarterMonitor[5]);
printf("\nThe number of words starting with G: %d\n", wordStarterMonitor[6]);
printf("\nThe number of words starting with H: %d\n", wordStarterMonitor[7]);
printf("\nThe number of words starting with I: %d\n", wordStarterMonitor[8]);
printf("\nThe number of words starting with J: %d\n", wordStarterMonitor[9]);
printf("\nThe number of words starting with K: %d\n", wordStarterMonitor[10]);
printf("\nThe number of words starting with L: %d\n", wordStarterMonitor[11]);
```

```
    printf("\nThe number of words starting with M: %d\n", wordStarterMonitor[12]);
    printf("\nThe number of words starting with N: %d\n", wordStarterMonitor[13]);
    printf("\nThe number of words starting with O: %d\n", wordStarterMonitor[14]);
    printf("\nThe number of words starting with P: %d\n", wordStarterMonitor[15]);
    printf("\nThe number of words starting with Q: %d\n", wordStarterMonitor[16]);
    printf("\nThe number of words starting with R: %d\n", wordStarterMonitor[17]);
    printf("\nThe number of words starting with S: %d\n", wordStarterMonitor[18]);
    printf("\nThe number of words starting with T: %d\n", wordStarterMonitor[19]);
    printf("\nThe number of words starting with U: %d\n", wordStarterMonitor[20]);
    printf("\nThe number of words starting with V: %d\n", wordStarterMonitor[21]);
    printf("\nThe number of words starting with W: %d\n", wordStarterMonitor[22]);
    printf("\nThe number of words starting with X: %d\n", wordStarterMonitor[23]);
    printf("\nThe number of words starting with Y: %d\n", wordStarterMonitor[24]);
    printf("\nThe number of words starting with Z: %d\n\n", wordStarterMonitor[25]);

    printf("\n\nThe letter that starts most words is : %c\n\n", mostWordStart);
    fclose(filePointer);
    exit(EXIT_SUCCESS);
}

// takes integer representation of a letter and returns the letter
char letterFromPosition(int position)
{
    char letter;
    switch (position)
    {
    case 0:
        letter = 'A';
        break;
    case 1:
        letter = 'B';
        break;
    case 2:
        letter = 'C';
        break;
    case 3:
        letter = 'D';
        break;
    case 4:
        letter = 'E';
        break;
    case 5:
        letter = 'F';
        break;
    case 6:
        letter = 'G';
        break;
    case 7:
        letter = 'H';
    break;
case 8:
    letter = 'I';
    break;
case 9:
    letter = 'J';
    break;
case 10:
    letter = 'K';
    break;
case 11:
    letter = 'L';
    break;
case 12:
    letter = 'M';
    break;
case 13:
    letter = 'N';
    break;
case 14:
    letter = 'O';
    break;
case 15:
    letter = 'P';
    break;
case 16:
    letter = 'Q';
    break;
case 17:
    letter = 'R';
    break;
case 18:
    letter = 'S';
    break;
case 19:
    letter = 'T';
    break;
case 20:
    letter = 'U';
    break;
case 21:
    letter = 'V';
    break;
case 22:
    letter = 'W';
    break;
case 23:
    letter = 'X';
    break;
    case 24:
        letter = 'Y';
        break;
    case 25:
        letter = 'Z';
        break;
    default:
        break;
    }
    return letter;
}
```