Introduction to LyX

Akshit Kumar (EE14B127)

11th August 2015

Abstract

This report contains the source code for the implementation of the C program which reads the text of a file and gives the number of words of different lengths and tabulates it in an horizontal histogram and outputs the data. The C program has been implemented using both arrays and pointes. In the array implementation array indexing is used to retrieve the value at that index. In the pointer implemenation a pointer is used to traverse the buffer array.

1 C Assignment

1.1 Implementation using arrays

The following code reads the contents of a file passed as command line arguments and calculates the number of words of different lengths. It does so by looping through the whole buffer array until it encounter a '\0'. Following is the code :

```
#include<stdio.h> // Library File Access
#include<stdlib.h> // Library File Access
#define MAXLENGTH 512 //Defining the MAXLENGTH of string to be taken
/* main() function takes argument from Command Line */
int main(int argc, char **argv){
    /*Program expects a filename. Check that argument was passed */
    if(argc!=2){
        printf("Usage ./a.out <filename>");
        exit(1);
    }
    /* Open file while checking for existence */
    FILE *fp = fopen(argv[1],"r");
    if(fp==NULL){
        printf("File could not be opened");
        exit(2);
    }
    /* Read in lines from file and process. Note the use of fgets() which is */
    /* more secure than gets() */
    char buf[MAXLENGTH]; // Declaring the array buf to store file stream
    int lengthFreq[11] = {0}; // Hold counting information and initialising to 0
    while(fgets(buf,MAXLENGTH,fp)){
        int count = 0,i=0; // Initialising iteration and counter variable
        /* Iterating through the buf array till we encounter the end of buf array */
        while(buf[i]!='\0'){
            /* Checking for word separators like ' ','\t',',' and end of buf array */
            if(buf[i]==' ' || buf[i]==',' || buf[i]==':' || buf[i]=='\t' ||
                buf[i]==';' || buf[i+1]=='\0'){
                lengthFreq[count]++;
                count = 0;
            }
            else{
                /* Checking for the alphabet */
                if((buf[i]>='A' && buf[i]<='Z') || (buf[i]>='a' && buf[i]<='z')){
                        count++;
                }
            }
        i++;
        }
}
```

```
    /* Printing out the result of the analysis */
    int k;
    for(k=3;k<=10;k++){
        printf("%d Letter Words: %d\n",k,lengthFreq[k]);
    }
    return 0;
}
```

1.2 Implementation using Pointers

The following code reads the contents of a file passed as command line arguments and calculates the number of words of different lengths. It does by making use of pointers which initially points to the zero indexed element of buf array and value at any index - i is retrieved by making use of a pointer ptr. Following is the code :

```
#include<stdio.h> // Library File Access
#include<stdlib.h> // Library File Access
#define MAXLENGTH 512 //Defining the MAXLENGTH of string to be taken
/* main() function takes argument from Command Line */
int main(int argc, char **argv){
    /*Program expects a filename. Check that argument was passed */
    if(argc!=2){
        printf("Usage ./a.out <filename>");
        exit(1);
    }
    /* Open file while checking for existence */
    FILE *fp = fopen(argv[1],"r");
    if(fp==NULL){
        printf("File could not be opened");
        exit(2);
    }
    /* Read in lines from file and process. Note the use of fgets() which is */
    /* more secure than gets() */
    char buf[MAXLENGTH]; // Array buf in which the file stream will be stored
    int lengthFreq[11] = {0}; // Hold counting information and initialising to 0
    while(fgets(buf,MAXLENGTH,fp)){
        int count = 0,i=0; // Initialising iteration and counter variable
        char *ptr; // Declaration of a pointer ptr
        ptr = buf; // ptr points to the address of buf[0]
        while(*ptr!='\0'){
            /* Checking for word separators like ' ','\t',',' and end of buf array */
            if(*ptr==' ' || *ptr==',' || *ptr==':' || *ptr=='\t' ||
             *ptr==';' || *(ptr+1)=='\0'){
                lengthFreq[count]++;
                count = 0;
```

```
            }
            else{
                /* Checking for the alphabet */
                if((*ptr>='A' && *ptr<='Z') || (*ptr>='a' && *ptr<='z')){
                    count++;
                }
            }
            ptr++;
        }
    }
    /* Printing out the result of the analysis */
    int k;
    for(k=3;k<=10;k++){
        printf("%d Letter Words: %d\n",k,lengthFreq[k]);
    }
    return 0;
}
```

1.3 Implementation of Horizontal Histogram - Extra Credit Assignment

1.3.1 Implemenation using arrays

The following code in addition to giving raw data, also prints out an histogram displaying the number of words of different length in a text file. This is implemented using arrays. Following is the code:

```
#include<stdio.h> // Library File Access
#include<stdlib.h> // Library File Access
#define MAXLENGTH 512 //Defining the MAXLENGTH of string to be taken
/* main() function takes argument from Command Line */
int main(int argc, char **argv){
    /*Program expects a filename. Check that argument was passed */
    if(argc!=2){
        printf("Usage ./a.out <filename>");
        exit(1);
    }
    /* Open file while checking for existence */
    FILE *fp = fopen(argv[1],"r");
    if(fp==NULL){
        printf("File could not be opened");
        exit(2);
    }
    /* Read in lines from file and process. Note the use of fgets() which is */
    /* more secure than gets() */
    char buf[MAXLENGTH]; // Array buf in which the file stream will be stored
```

```
    int lengthFreq[11] = {0}; // Hold counting information and initialising to 0
    while(fgets(buf,MAXLENGTH,fp)){
        int count = 0,i=0; // Initialising iteration and counter variable
        /* Iterating through the buf array till we encounter the end of buf array */
        while(buf[i]!='\0'){
            /* Checking for word separators like ' ','\t',',' and end of buf array */
            if(buf[i]==' ' || buf[i]==',' || buf[i]==':' || buf[i]=='\t' ||
                buf[i]==';' || buf[i+1]=='\0'){
                lengthFreq[count]++;
                count = 0;
            }
            else{
                /* Checking for the alphabet */
                if((buf[i]>='A' && buf[i]<='Z') || (buf[i]>='a' && buf[i]<='z')){
                    count++;
                }
            }
            i++;
        }
    }
    /* Printing out the result of the analysis */
    int k,j;
    for(k=3;k<=10;k++){
        printf("%d Letter Words: %d\n",k,lengthFreq[k]);
    }
    /* Printing of the horizontal histogram */
    printf("\n");
    /* Printing the line for 10 separately */
    printf("10|");
    for(j=0;j<lengthFreq[10];j++){
        printf(" * ");
    }
    printf("\n");
    /* Nested loop to print the frequency of different word lengths */
    for(k=9;k>=1;k--){
        printf("%d | ",k);
        for(j=0;j<lengthFreq[k];j++){
            printf("* ");
        }
        printf("\n");
    }
    printf("0 +---------------------------------- \n");
    return 0;
}
```

1.3.2 Implementation using pointers

The following code in addition to giving raw data, also prints out an histogram displaying the number of words of different length in a text file. This is implemented using pointers. Following is the code:

```
#include<stdio.h> // Library File Access
#include<stdlib.h> // Library File Access
#define MAXLENGTH 512 //Defining the MAXLENGTH of string to be taken
/* main() function takes argument from Command Line */
int main(int argc, char **argv){
    /*Program expects a filename. Check that argument was passed */
    if(argc!=2){
        printf("Usage ./a.out <filename>");
        exit(1);
    }
    /* Open file while checking for existence */
    FILE *fp = fopen(argv[1],"r");
    if(fp==NULL){
        printf("File could not be opened");
        exit(2);
    }
    /* Read in lines from file and process. Note the use of fgets() which is */
    /* more secure than gets() */
    char buf[MAXLENGTH]; // Array buf in which the file stream will be stored
    int lengthFreq[11] = {0}; // Hold counting information and initialising to 0
    while(fgets(buf,MAXLENGTH,fp)){
        int count = 0,i=0; // Initialising iteration and counter variable
        char *ptr; // Declaration of a pointer ptr
        ptr = buf; // ptr points to the address of buf[0]
        while(*ptr!='\0'){
            /* Checking for word separators like ' ','\t',',' and end of buf array */
            if(*ptr==' ' || *ptr==',' || *ptr==':' || *ptr=='\t' ||
             *ptr==';' || *(ptr+1)=='\0'){
                lengthFreq[count]++;
                count = 0;
            }
            else{
                /* Checking for the alphabet */
                if((*ptr>='A' && *ptr<='Z') || (*ptr>='a' && *ptr<='z')){
                    count++;
                }
            }
            ptr++;
        }
    }
    /* Printing out the result of the analysis */
```

```
    int k,j;
    for(k=3;k<=10;k++){
        printf("%d Letter Words: %d\n",k,lengthFreq[k]);
    }
    /* Printing of the horizontal histogram */
    printf("\n");
    /* Printing the line for 10 separately */
    printf("10|");
    for(j=0;j<lengthFreq[10];j++){
        printf(" * ");
    }
    printf("\n");
    /* Nested loop to print the frequency of different word lengths */
    for(k=9;k>=1;k--){
        printf("%d | ",k);
        for(j=0;j<lengthFreq[k];j++){
            printf("* ");
        }
        printf("\n");
    }
    printf("0 +---------------------------------- \n");
    return 0;
}
```

2 Output of the Programs

2.1 Implementation using Arrays

For the test.txt file

Hello, this is a sample document which contains some random text.
Some random words are : Apple , Ball , Cat , Dog , Egg, Fish , Goat
, Hen etc.

OUTPUT is :

```
3 Letter Words: 6
4 Letter Words: 7
5 Letter Words: 4
6 Letter Words: 3
7 Letter Words: 0
8 Letter Words: 2
9 Letter Words: 0
10 Letter Words: 0
```

2.2 Implementation using Pointers

For the test.txt file

Hello, this is a sample document which contains some random text. Some random words are : Apple , Ball , Cat , Dog , Egg, Fish , Goat , Hen etc.

OUTPUT is :

```
3 Letter Words: 6
4 Letter Words: 7
5 Letter Words: 4
6 Letter Words: 3
7 Letter Words: 0
8 Letter Words: 2
9 Letter Words: 0
10 Letter Words: 0
```

2.3 Extra Credit Assignment

For the test.txt file

Hello, this is a sample document which contains some random text. Some random words are : Apple , Ball , Cat , Dog , Egg, Fish , Goat , Hen etc.

OUTPUT is :

```
3 Letter Words: 6
4 Letter Words: 7
5 Letter Words: 4
6 Letter Words: 3
7 Letter Words: 0
8 Letter Words: 2
9 Letter Words: 0
10 Letter Words: 0
10|
9 |
8 | * *
7 |
6 | * * *
5 | * * * *
4 | * * * * * * *
3 | * * * * * *
2 | *
1 | *
0 +---------------------------------
```