Lab 5

String Sorting and Characters Counting

Outline

- In Brightspace you are given 1 example program: *insertionSort.c*
  - Download from Brightspace the insertionSort.c program.
  - Open CLion and create a new project called Sort.
  - Create a new C Source file called: insertionSort.c.
  - Copy and paste the code contained in the downloaded file: insertionSort.c
  - Run the program and understand the code by reading the comments and the slides.
  - This example is useful to tackle Assignment 1

1. *insertionSort.c*
   - It is a program that sorts 5 strings provided as input from the command line

Outline

- In Brightspace you are given 1 example program: *charCount.c*.
  - Download from Brightspace the charCount.c program.
  - Create a new project in CLion.
  - Create a new C Source file called: charCount.c.
  - Copy and paste the code contained in the downloaded file: charCount.c
  - Run the program and understand the code by reading the comments and the slides.
  - These examples are useful to tackle Assignment 1

2. *charCount.c*
   - It is a module that counts the number of occurrences of each character in the string.

insertionSort.c

insertionSort.c

```
int main() {
    char inputStrings[STR_NUM][MAX_LEN];
    insertStrings(inputStrings);
    insertionSort(inputStrings);
    printStrings(inputStrings);

}
```

- **insertStrings:**
  - reads 5 strings from the standard input and writes them in *inputStrings*
- **insertionSort:**
  - Sorts the strings in *inputStrings* in alphabetical order using the insertion sort algorithm
- **printStrings:**
  - Prints the strings stored in *inputStrings*

insertStrings

```c
/*
 * This function asks the user to input 5 strings
 * and writes them in the array "inputStrings"
 */
void insertStrings(char inputStrings [][MAX_LEN]){
    printf("Insert %d Strings as input\n", STR_NUM);
    for(int i =0; i < STR_NUM; i++){
        printf("String %d: ", i+1);

        //reads strings from the standard input (stdin)
        fgets(inputStrings[i],MAX_LEN, stdin);

        //removes the carriage return character from the input strings
        if(inputStrings[i][strlen(inputStrings[i])-1] == '\n'){
            inputStrings[i][strlen(inputStrings[i])-1] = '\0';
        }
    }
}
```

Asks the user to input 5 strings and writes them in the 2D array “inputStrings”

insertStrings

```c
/*
 * This function asks the user to input 5 strings
 * and writes them in the array "inputStrings"
 */
void insertStrings(char inputStrings [][MAX_LEN]){
    printf("Insert %d Strings as input\n", STR_NUM);
    for(int i =0; i < STR_NUM; i++){
        printf("String %d: ", i+1);

        //reads strings from the standard input (stdin)
        fgets(inputStrings[i],MAX_LEN, stdin);

        //removes the carriage return character from the input strings
        if(inputStrings[i][strlen(inputStrings[i])-1] == '\n'){
            inputStrings[i][strlen(inputStrings[i])-1] = '\0';
        }
    }
}
```

- Function **fgets** is used to read from the standard input (*stdin*)
- To refer to variable *stdin*, you need to include library stdlib.h

insertStrings

```c
/*
 * This function asks the user to input 5 strings
 * and writes them in the array "inputStrings"
 */
void insertStrings(char inputStrings [][MAX_LEN]){
    printf("Insert %d Strings as input\n", STR_NUM);
    for(int i =0; i < STR_NUM; i++){
        printf("String %d: ", i+1);

        //reads strings from the standard input (stdin)
        fgets(inputStrings[i],MAX_LEN, stdin);

        //removes the carriage return character from the input strings
        if(inputStrings[i][strlen(inputStrings[i])-1] == '\n'){
            inputStrings[i][strlen(inputStrings[i])-1] = '\0';
        }
    }
}
```

It is important to remove the ‘\n’ character at the end of the string to avoid problems during printing and sorting.

toLower

```
/*
 * This function converts inputString in a string having all
 * lowercase letters and copies the result in lowerString
 */
void toLower(char inputString[], char lowerString[]){

    for(int i = 0; i< strlen(inputString); i++){
        // function to lower converts a character in its lower case version
        lowerString[i] = tolower(inputString[i]);
    }
}
```

Converts a string provided as input to lowercase characters and writes the result into *lowerString*

toLower

```
/*
 * This function converts inputString in a string having all
 * lowercase letters and copies the result in lowerString
 */
void toLower(char inputString[], char lowerString[]){

    for(int i = 0; i< strlen(inputString); i++){
        // function to lower converts a character in its lower case version
        lowerString[i] = tolower(inputString[i]);
    }
}
```

- For each character in the string it applies the C function *toLower* that converts each character into lower case letters.

- To use C function *toLower* you need to include library **ctype.h**

insertionSort

```
void insertionSort(char inputStrings [][MAX_LEN]) {
    int i, j;

    char swap[MAX_LEN], lowerString1[MAX_LEN], lowerString2[MAX_LEN];
    for (i = 1; i < STR_NUM; i++) {
        j = i;

        // before comparing strings at positions j and j-1
        // it is necessary to convert them into lowercase strings
        toLower(inputStrings[j], lowerString1);
        toLower( inputString: inputStrings[j-1], lowerString2);

        while (j > 0 && strcmp(lowerString1, lowerString2) < 0) {
            // swapping of strings is performed using strcpy
            strcpy(swap, inputStrings[j]);
            strcpy(inputStrings[j], inputStrings[j - 1]);
            strcpy(inputStrings[j - 1], swap);
            j--;
            toLower(inputStrings[j], lowerString1);
            toLower( inputString: inputStrings[j-1], lowerString2);
        }
    }
}
```

Sorts the strings provided as input

insertionSort

```
void insertionSort(char inputStrings [][MAX_LEN]) {
    int i, j;

    char swap[MAX_LEN], lowerString1[MAX_LEN], lowerString2[MAX_LEN];
    for (i = 1; i < STR_NUM; i++) {
        j = i;

        // before comparing strings at positions j and j-1
        // it is necessary to convert them into lowercase strings
        toLower(inputStrings[j], lowerString1);
        toLower( inputString: inputStrings[j-1], lowerString2);

        while (j > 0 && strcmp(lowerString1, lowerString2) < 0) {
            // swapping of strings is performed using strcpy
            strcpy(swap, inputStrings[j]);
            strcpy(inputStrings[j], inputStrings[j - 1]);
            strcpy(inputStrings[j - 1], swap);
            j--;
            toLower(inputStrings[j], lowerString1);
            toLower( inputString: inputStrings[j-1], lowerString2);
        }
    }
}
```

Before comparing strings, it converts them to lower case letters.

insertionSort

```
void insertionSort(char inputStrings [][MAX_LEN]) {
    int i, j;

    char swap[MAX_LEN], lowerString1[MAX_LEN], lowerString2[MAX_LEN];
    for (i = 1; i < STR_NUM; i++) {
        j = i;

        // before comparing strings at positions j and j-1
        // it is necessary to convert them into lowercase strings
        toLower(inputStrings[j], lowerString1);
        toLower( inputString: inputStrings[j-1], lowerString2);

        while (j > 0 && strcmp(lowerString1, lowerString2) < 0) {
            // swapping of strings is performed using strcpy
            strcpy(swap, inputStrings[j]);
            strcpy(inputStrings[j], inputStrings[j - 1]);
            strcpy(inputStrings[j - 1], swap);
            j--;
            toLower(inputStrings[j], lowerString1);
            toLower( inputString: inputStrings[j-1], lowerString2);
        }
    }
}
```

Strings swapping is performed using function strcpy

insertionSort

```
void insertionSort(char inputStrings [][MAX_LEN]) {
    int i, j;

    char swap[MAX_LEN], lowerString1[MAX_LEN], lowerString2[MAX_LEN];
    for (i = 1; i < STR_NUM; i++) {
        j = i;

        // before comparing strings at positions j and j-1
        // it is necessary to convert them into lowercase strings
        toLower(inputStrings[j], lowerString1);
        toLower( inputString: inputStrings[j-1], lowerString2);

        while (j > 0 && strcmp(lowerString1, lowerString2) < 0) {
            // swapping of strings is performed using strcpy
            strcpy(swap, inputStrings[j]);
            strcpy(inputStrings[j], inputStrings[j - 1]);
            strcpy(inputStrings[j - 1], swap);
            j--;
            toLower(inputStrings[j], lowerString1);
            toLower( inputString: inputStrings[j-1], lowerString2);
        }
    }
}
```

Decrements j and converts the strings to lower case letters before comparing them in the next cycle of the while loop.

printStrings

```
/*
 * This function prints a set of strings provided as input
 */
    void printStrings(char inputStrings [][MAX_LEN]){
        printf("The list of ordered strings is:\n");
        for(int i =0; i< STR_NUM; i++)
            printf("%s\n",inputStrings[i] );
    }
```

Prints the sorted array of strings.

charCount.c

charCount.c

```
int main() {
    //2D array storing the strings provided as input
    char inputStrings[STR_NUM][MAX_LEN];
    //2D array storing information about the number of characters
    int charCount[STR_NUM][CHAR_NUM];

    insertStrings(inputStrings);
    countCharacters(inputStrings, charCount);
    printCharCount(charCount);
}
```

- **inputStrings**
  - A 2Dimensional array of characters storing the strings provided as input
- **charCount**
  - A 2Dimensional array of integers storing the number of characters contained in the strings in *inputStrings*

charCount.c

```c
int main() {
    //2D array storing the strings provided as input
    char inputStrings[STR_NUM][MAX_LEN];
    //2D array storing information about the number of characters
    int charCount[STR_NUM][CHAR_NUM];

    insertStrings(inputStrings);
    countCharacters(inputStrings, charCount);
    printCharCount(charCount);
}
```

- **insertStrings:**
  - reads 5 strings from the standard input and writes them in *inputStrings.*
- **countCharacters:**
  - Counts the number of characters for the strings in *inputStrings* and stores the results in *charCoun*t.
- **printCharCount:**
  - Prints the number of characters in the sentences in *inputStrings*

countCharacters

```
void countCharacters(char inputStrings [][MAX_LEN], int charCount[][CHAR_NUM]) {
    char lowerString[MAX_LEN];

    //initialize charCount
    for(int i =0; i < STR_NUM; i++)
        for(int j =0;  j< CHAR_NUM; j++)
            charCount[i][j] = 0;

    for(int i =0; i< STR_NUM; i++){
        //converts a string into lower case letters
        toLower(inputStrings[i], lowerString);
        for (int j =0; j < strlen(inputStrings[i]); j++){
            //increments the cell in char count associated with the corresponding character
            switch(lowerString[j]){
                case 'a': charCount[i][0]++;
                          break;
                case 'b': charCount[i][1]++;
                          break;
                case 'c': charCount[i][2]++;
                    break;
                case 'd': charCount[i][3]++;
                    break;
```

countCharacters

```
void countCharacters(char inputStrings [][MAX_LEN], int charCount[][CHAR_NUM]) {
    char lowerString[MAX_LEN];

    //initialize charCount
    for(int i =0; i < STR_NUM; i++)
        for(int j =0;  j< CHAR_NUM; j++)
            charCount[i][j] = 0;

    for(int i =0; i< STR_NUM; i++){
        //converts a string into lower case letters
        toLower(inputStrings[i], lowerString);
        for (int j =0; j < strlen(inputStrings[i]); j++){
            //increments the cell in char count associated with the corresponding character
            switch(lowerString[j]){
                case 'a': charCount[i][0]++;
                          break;
                case 'b': charCount[i][1]++;
                          break;
                case 'c': charCount[i][2]++;
                    break;
                case 'd': charCount[i][3]++;
                    break;
```

Initialize the 2D array of integers storing the characters count.

countCharacters

```
void countCharacters(char inputStrings [][MAX_LEN], int charCount[][CHAR_NUM]) {
    char lowerString[MAX_LEN];

    //initialize charCount
    for(int i =0; i < STR_NUM; i++)
        for(int j =0;  j< CHAR_NUM; j++)
            charCount[i][j] = 0;

    for(int i =0; i< STR_NUM; i++){
        //converts a string into lower case letters
        toLower(inputStrings[i], lowerString);
        for (int j =0; j < strlen(inputStrings[i]); j++){
            //increments the cell in char count associated with the corresponding character
            switch(lowerString[j]){
                case 'a': charCount[i][0]++;
                          break;
                case 'b': charCount[i][1]++;
                          break;
                case 'c': charCount[i][2]++;
                    break;
                case 'd': charCount[i][3]++;
                    break;
```

Converts a string into lower case characters

countCharacters

```
void countCharacters(char inputStrings [][MAX_LEN], int charCount[][CHAR_NUM]) {
    char lowerString[MAX_LEN];

    //initialize charCount
    for(int i =0; i < STR_NUM; i++)
        for(int j =0;  j< CHAR_NUM; j++)
            charCount[i][j] = 0;

    for(int i =0; i< STR_NUM; i++){
        //converts a string into lower case letters
        toLower(inputStrings[i], lowerString);
        for (int j =0; j < strlen(inputStrings[i]); j++){
            //increments the cell in char count associated with the corresponding character
            switch(lowerString[j]){
                case 'a': charCount[i][0]++;
                          break;
                case 'b': charCount[i][1]++;
                          break;
                case 'c': charCount[i][2]++;
                    break;
                case 'd': charCount[i][3]++;
                    break;
```

Increments an element of the array corresponding to the row of the string considered and the column of the character encountered.

countCharacters

```
void printCharCount(int charCount [][CHAR_NUM]) {
    printf("The number of chanracters is:\n");
    for (int i = 0; i < STR_NUM; i++) {
        for (int j = 0; j < CHAR_NUM; j++)
            printf("%d ", charCount[i][j]);
    printf("\n");
    }
}
```

Prints the number of characters in the string provided as input (stored in *charCount*).