Computer Engineering Department

TED University

TED UNIVERSITY

CMPE 252 – C Programming, Spring 2023

Lab 03

In this lab, you are asked to complete **intersection.c** program file which has been already given in Moodle. In this program, there are three functions, namely, `main`, `findIntersection`, and `printStudent`. `main` function is already provided and it is supposed to remain as it is (you should not change it). You are required to implement `findIntersection` and `printStudent` functions.

Here are the operations performed in `main` function:

- An array of strings with name `list` is created to hold name, surname and grade data in "name_surname_grade" format.
- Two *arrays of pointers to strings* with names `group1` and `group2` are created along with two integer variables n1 and n2 to hold the number of elements in the arrays.
- n1 and `group1` are initialized by taking data from the standard input.
- n2 and `group2` are initialized by taking data from the standard input.
- *An array of pointers to strings* with name `common` and an integer variable with name `commonCount` are created. The array of pointers `common` is supposed to hold pointers to the strings that are included both in `group1` and in `group2`. `commonCount` is supposed to hold the number of elements in the intersection set. Those variables are to be filled by calling `findIntersection` function.
- `findIntersection` function is called with 6 arguments, which are the array of pointers (`group1` and `group2`), their number of elements (`n1` and `n2`), the array of pointers `common`, and the address of `commonCount` variable.
- The strings pointed to, by the elements of the array `group1` are printed.
- The strings pointed to, by the elements of the array `group2` are printed.
- The strings pointed to, by the elements of the array `common` are printed.
- The user is asked whether to print the name part of the strings in the intersection set in NamE SurnamE Grade format. If the answer is 1, then:
    - `printStudent` function is called with 2 arguments, which are the array of pointers `common` and its number of elements `commonCount`.
    - The strings pointed to, by the elements of the array `common` are printed again to check whether they remain unchanged after calling `printStudent` function.

Task 1: Implement `findIntersection` function.

```
void findIntersection(char *group1[], char *group2[], int n1, int n2, char *common[],
int *commonCountPtr);
```

The array of pointers `common` is an output parameter and it is supposed to hold pointers to the set of strings pointed to, both by the elements of `group1` and by the elements of `group2`. The total number of elements in the intersection is to be stored in the integer variable pointed to, by `commonCountPtr` (which is also an output parameter).

Hint: You need to compare each string in `group1` with each string in `group2`. You need to use `strcmp` function for the comparison.

```
Number of elements in group1: 6
Entries in group1: 1 4 9 3 2 8

Number of elements in group2: 4
Entries in group2: 9 5 1 6

group1:
zoe_lang_85
david_studi_75
dustin_vin_35
jack_alonso_50
sam_rodriguez_70
michael_loaf_45

group2:
dustin_vin_35
denzel_feldman_90
zoe_lang_85
james_bale_70

intersection of group1 and group2:
zoe_lang_85
dustin_vin_35

Do you want to print students in the intersection set in NamE SurnamE Grade format (1/0)? 0
```

Consider the above example run:

Suppose that group1 is initialized to have 6 elements, namely, list[1], list[4], list[9], list[3], list[2], and list[8]. group2 is initialized to have 4 elements, namely, list[9], list[5], list[1], and list[6]. As shown in Figure 1, the intersection of group1 and group2 according to grade contains list[1] and list[9].

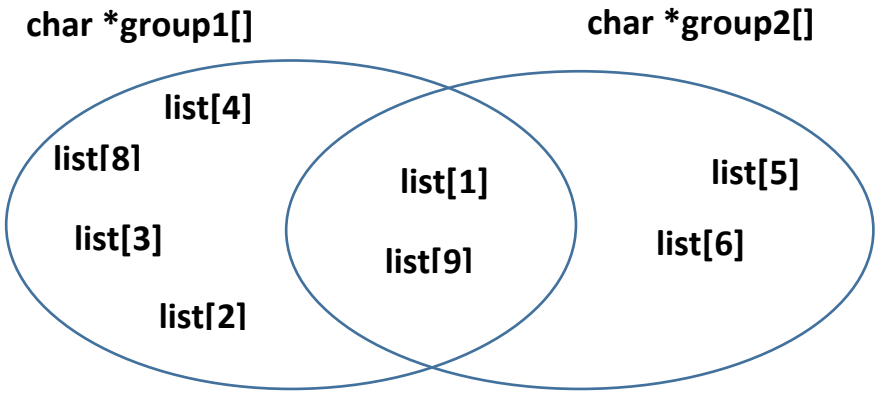

Figure 1: sets of strings

Task 2: Implement printStudent function.

```
void printStudent(char *common[], int commonCount);
```

NamE, SurnamE and Grade of students are printed where the first letter and the last letter of name and surname are capitalized. commonCount holds the number of elements in the array common.

Some functions that you might need to use are as follows:

```
//Copies the string pointed to, by src to dest.
strcpy(char* dest, const char* src);

//Breaks string str into a series of tokens separated by delim
strtok(char* str, const char* delim);

//Converts lowercase letter of str string to uppercase.
toupper(str[i]);
```

Hints:

strtok function changes the content of its first string argument so you need to call strtok function on a copy of the string argument if you do not want to change its content. For each string that can be accessed using common, you should copy it into a different memory space. Consider to create a local char array (string) with size STR_LEN and use it for that purpose.

```
Number of elements in group1: 6
Entries in group1: 1 4 9 3 2 8

Number of elements in group2: 5
Entries in group2: 9 5 1 6 2

group1:
zoe_lang_85
david_studi_75
dustin_vin_35
jack_alonso_50
sam_rodriguez_70
michael_loaf_45

group2:
dustin_vin_35
denzel_feldman_90
zoe_lang_85
james_bale_70
sam_rodriguez_70

intersection of group1 and group2:
zoe_lang_85
dustin_vin_35
sam_rodriguez_70

Do you want to print students in the intersection set in NamE SurnamE Grade format
(1/0)? 1

Students in the intersection of group1 and group2:
ZoE LanG 85
DustiN ViN 35
SaM RodrigueZ 70

intersection of group1 and group2:
zoe_lang_85
dustin_vin_35
sam_rodriguez_70
```

Above is an example run: