BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI (RAJ.)

Second Semester 2019-20 CS F111 Computer Programming

LABORATORY SESSION #11

(Structures)

1. The following details of students are made available in a file (as comma separated fields):

ID number, name, gender (M or F), age (a whole number), residential status (H for hostel resident, D for day scholar), and CGPA.

Two sample rows (i.e., two student records) of the data file (data.txt) are shown below:

```
2014A7PS0108P,Indra Gopinath,F,21,H,8.55
2017B1TS1055P,Mohammed Farhan,M,18,D,0.0
```

Write a *well-commented* C program, made modular using user-defined functions, to accomplish the tasks listed below. You may copy the file `/home/share/stud_records.c` into your current folder. The `main()` function with function calls, function prototypes as well as definitions are all there. The data file is available at `/home/share/stud_records.txt` which you can use.

(a) Read each student record from the data file, and store in an array of structures, each element of the array representing a record. Write a function with the following prototype to do this task.

```
void populateRecords(struct studsarr[], int no_studs);
                        /* function to get data into the array */
```

(b) The complete definition of the function for generating the email address (for a given student ID number) has been provided to you.

```
void generateEmailAddress(char idno[], char email[]);
                        /* function definition already given */
```

Read the function definition and learn how it works, and also how to write meaningful comments for code. Also note the use of the function **`atoi()`** in the function. You can look up the manual of this function to know more about it.

Now write the following function that calls `generateEmailAddress()` and then stores the email address in the right field of each student record of the array:

```
void storeEmailAddresses(struct studsarr[], int no_studs);
```

(c) Print out details of all students by writing a function whose prototype is:

```
void printRecords(struct studs [], int);
```

Formatted output should appear as shown below (and in the next page):

```
2014A7PS0108P Indra Gopinath
Gender: F
Age: 21
Residence status: Hostel
CGPA: 8.55
Email: f2014108@pilani.bits-pilani.ac.in
-----------------------------------------------
```

```
2017B1TS1055P Mohammed Farhan

Gender: M

Age: 18

Residence status: Day scholar

CGPA: Not available

Email: f20171055@pilani.bits-pilani.ac.in

-----------------------------------------------
```

(d) Write a function to compute the average CGPA of all CGPAs that are available, and to return that a value back to `main()`:

```
float calculateAvgCG(STUD *, int);
```

2. "Do Indian parents prefer giving shorter names over longer ones to their children?" You will write a *well-commented* and modular C program to help get a possible answer to this question, using data from a given sample population. You will use the following structure definition in your program:

```
typedef struct {
    char fname[70];    /* first name (a single word) */
    int freq; /* how many people in the sample population have this name
    */ int len; /* stores the length of the name */
} NAME;
```

Given a sample size of **N** individuals, and a positive integer **threshold** denoting a length, your program should do these tasks:

(a) Obtain data for **N** individuals and store it appropriately using the function:

```
void getData(NAME arr[]);
```

(b) Calculate and print the percentage of people in the sample population whose first names are shorter than or equal to the threshold length, using another function:

```
float findPercent(NAME arr[], int threshold);
```

The file `/home/share/names.txt` available on *Prithvi* has sample data – each line (except the first one) containing a unique first name and the frequency of its occurrence in the sample population. For example, in the sample data shown below, there are six people with the name Abhinav, four with Abhimanyu, two with Abhay and one each of Abhas, Abhay, Abhijeet and Abhijit:

```
ABHAS          1
ABHAY          2
ABHIJEET       1
ABHIJIT        1
ABHIMANYU      4
ABHINAV        6
```

The first line of the file contains two numbers – the number of unique names (stored one per row) and the value for `threshold`.