Id. No.:	Name:

B

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

First Semester 2017-18 CS F111 Computer Programming

18-Nov-2017 — ONLINE TEST (SOLUTIONS) — 30 marks (15%), 1½ hrs.

1. You are given a `ROWS` x `COLS` integer matrix called **mat**, and another array called **shift** of `ROWS` numbers. (`ROWS` and `COLS` are `#define`d constants). You have to circularly shift the numbers of each row of **mat** to the *left* by the number of positions mentioned in corresponding value in the **shift** array. Your C program should print the modified matrix along with column-wise totals. Here is an illustrative example:

```
          mat                       shift                         mat
3     4     -2    16                  1                       4    -2    16    3
12    76    0     5                   4    after left shifting  12   76    0     5
16    7     100   -70                 6    ------------------>  100  -70   16    7
                                                                ---------------------
                                                                116   4    32    15
                                                                ---------------------
```

Note that the shift operation is not valid for negative numbers (which means negative input in **shift** array should be appropriately dealt with). Your program should be commented well, written modularly, and use the following functions that you have to define:

`void populateMatrix(int [][COLS]);`	[3]
`void leftShiftRow(int *);`	[6]
`void printMatrixAndColTotals(int [][COLS]);`	[5]

```c
#include <stdio.h>
#define ROWS 3
#define COLS 4
void populateMatrix(int [][COLS]);
void leftShiftRow(int *, int);
void printMatrixAndColTotals(int [][COLS]);

int main() {
int mat[ROWS][COLS], shift[ROWS], i, j;

populateMatrix(mat);
for (i = 0; i < ROWS; ++i)
  do {
        printf("Enter values for the shift matrix (+ve values only): ");
        scanf("%d",&shift[i]);
     } while (shift[i] < 0);

for (i = 0; i < ROWS; ++i)
  leftShiftRow(mat[i],shift[i]);
printf("\nThe matrix after right shifting the rows is:\n\n");
printMatrixAndColTotals(mat);
} /* end of main() */

void populateMatrix(int a[][COLS]) {
int i, j;
for (i = 0; i < ROWS; ++i)
  for (j = 0; j < COLS; ++j)
    scanf("%d",&a[i][j]);
}
```

```
void leftShiftRow(int *row, int val) {
int temp[COLS], i;
for (i = 0; i < COLS; ++i)
  if (i- (val % COLS) >= 0)
        temp[i- (val % COLS)] = row[i];
  else /* negative indices wraparound */
        temp[COLS + i - (val % COLS)] = row[i];
for (i = 0; i < COLS; ++i)
   row[i] = temp[i];
}

void printMatrixAndColTotals(int mat[][COLS]) {
int i, j;
int coltotals[COLS] = {0};
for (i = 0; i < ROWS; ++i)
{
  putchar('\n');
  for (j = 0; j < COLS; ++j)
    printf("%3d\t",mat[i][j]);
  putchar('\n');
}

for (j = 0; j < COLS; ++j)
  for (i = 0; i < ROWS; ++i)
    coltotals[j] += mat[i][j]; /* summing up the columns */

for (i = 1; i <= COLS * 7; ++i)
  putchar('-');
putchar('\n');

for (j = 0; j < COLS; ++j)
   printf("%3d\t",coltotals[j]);
putchar('\n');

for (i = 1; i <= COLS * 7; ++i)
  putchar('-');
putchar('\n');
putchar('\n');
}
```

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[MAX];  /* 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**, and a positive integer **threshold** denoting a length, your program will:

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

```
void getData(NAME *);
```

[5]

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

`float findPercent(NAME *, int);` [8]

The file **names.txt** available in `/home/share` has sample data – each line (except the first one) containing a first name and the frequency of its occurrence in the sample population. The first line of the file contains two numbers – the sample size **N** and the value of **threshold**.

You are also required to turn over this page and write down the overall logic you will use and the layout of this second program, and return this sheet to the invigilator when you finish the test. Do not write any C code overleaf! [3]

```
#include <stdio.h>
#include <string.h>
#define MAX 50
#define STUDS 674

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

void getData(NAME data[]);
float findPercent(NAME data[],int);
int N;

int main() {
NAME data[STUDS];
int i, threshold;
scanf("%d%d", &N, &threshold);
getData(data);
printf("Percentage whose names are shorter than %d letters: %.2f\n", threshold,
      findPercent(data,threshold));
}

void getData(NAME data[]) {
int i;
for (i = 0; i < N; ++i)
   {
   scanf("%s%d",data[i].fname,&data[i].freq);
   data[i].len = strlen(data[i].fname);   /* storing the length of name */
   }
}

float findPercent(NAME data[], int threshold) {
int i;
float sum = 0.0, total = 0.0;
for (i = 0; i < STUDS; ++i)
   {
   total += data[i].freq; /* total number of students */
   if (data[i].len <= threshold) sum += data[i].freq; /* those who have less than
                              the threshold length */
   }
return (sum * 100/total);
}
```