Princess Sumaya University for Technology
جامعة الأميرة سميّة للتكنولوجيا

The King Hussein School for Computing Sciences
Department of Computer Science
Structured Programming - Spring 2023

Second Exam

Full Name: **Student ID:**

Question	Points	Score
1	PART 1.A: 4	
	PART 1.B: 4	
	PART 2: 3	
2	PART 1: 4	
	PART 2: 4	
3	6	
Total	25	

Circle your section:

- ○ Dr. Ammar Alrashdan (section 1)
- ○ Dr. Osama Alhaj Hasan (section 2)
- ○ Dr. Rawan Ghnemat (section 3)
- ○ Dr. Ammar Alrashdan (section 4)
- ○ Dr. Rawan Ghnemat (section 5)
- ○ Dr. Mohammad Al Nabhan (section 6)
- ○ Dr. Mohammad Al Nabhan (section 7)
- ○ Manaf Gharaibeh (section 8)
- ○ Dr. Mohammad Abu Snober (section 9)
- ○ Dr. Mohammad Abu Snober (section 10)
- ○ Mr. Yousef Yaseen (section 11)
- ○ Mr. Alaa Altarazi (section 12)
- ○ Mr. Alaa Altarazi (section 13)
- ○ Mr. Alaa Altarazi (section 14)

Monday 15/5/2023

Question 1 (11 points)

PART 1. Implement each of the following functions (assume that `N` is a globally defined constant).

A. `[4 points]` Function **`identity`**`(...)` receives a `2D` array of integers of size `NxN` and returns `1` if the main diagonal is all `1`s and all the other elements are zeroes (and returns `0` otherwise).

Examples.

```
1

1 0
0 1

1 0 0
0 1 0
0 0 1

1 0 0 0
0 1 0 0
0 0 1 0
0 0 0 1
```

These are all identity matrices

```
int identity(int a[][N], int N) {
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            if ((i == j && a[i][j] != 1) || (i != j && a[i][j] != 0))
                return 0;
        }
    }
    return 1;
}
```

B. `[4 points]` Function **`shift`**`(...)` receives a `2D` array of size `NxN` and shifts all the rows one position down. The first row becomes all zeroes and the last row is lost.

Example.

```
1 1 1 1
2 2 2 2
3 3 3 3
```

becomes

```
0 0 0 0
1 1 1 1
2 2 2 2
```

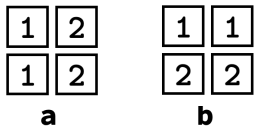
```
void shift(int a[][N]) {
    for (int i = N-1; i > 0; i--) {
        for (int j = 0; j < N; j++) {
            a[i][j] = a[i-1][j];
        }
    }

    for (int j = 0; j < N; j++) {
        a[0][j] = 0;
    }
}
```

PART 2. Answer the question below assuming that a, b and c are 2D arrays of size NxN.

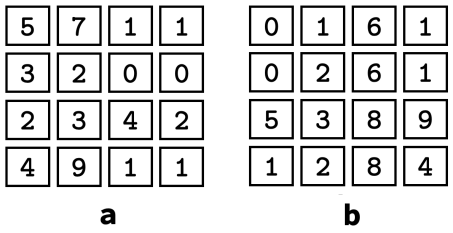
```
for (int i = 0; i < N; i++) {
      for (int j = 0; j < N; j++) {
            int sum = 0;
            for (int k = 0; k < N; k++) {
                  sum += a[i][k] * b[k][j];
            }
            c[i][j] = sum;
      }
}
```

A. [1 points] What will be stored at c[0][0] after the code finishes execution if a[][] and b[][] are shown on the right?



Answer = **1 + 4 = 5**

B. [2 points] What will be stored at c[2][1] after the code finishes execution if a[][] and b[][] are shown on the right?



Answer = **2 + 6 + 12 + 4 = 24**

Question 2 (8 points)

PART 1. [4 points] Convert the iterative function shown below to a recursive function and then show how your recursive function can be called. You are allowed to add or remove parameters.

```
int f1(int n, int a[], int flag) {
  if (n < 0) return 0;
  if (flag) return a[n] + f1(n-1, a, 0);
  else      return f1(n-1, a, 1);
}

// A better solution:
int f1(int n, int a[]) {
    if (n < 0) return 0;
    return a[n] + f1(n-2, a);
}
// called as follows:
// int result = f1(n-1, a);
```

```
int f1(int n, int a[]) {
  int sum = 0;
  int flag = 1;
  for (int i = n-1; i >= 0; i--) {
      if (flag == 1)
          sum += a[i];
      flag *= -1;
  }
  return sum;
}
```

Show how your function can be called on an array named a of size n.

```
int result = f1(n-1, a, 1);
```

PART 2. [4 points] What is the output of the following program?

```
int trace1(int n1, int n2) {
    if (n1 < n2) return 0;
    return 1 + trace1(n1 - n2, n2);
}

int trace2(int n) {
    if (n <= 0) return 0;
    return n + trace2(n - 1)
             + trace2(n - 1);
}

int main() {
    printf("%d\n", trace1(4, 1));
    printf("%d\n", trace1(55000, 5));
    printf("%d\n", trace2(2));
    printf("%d\n", trace2(4));
    return 0;
}
```

4

11000

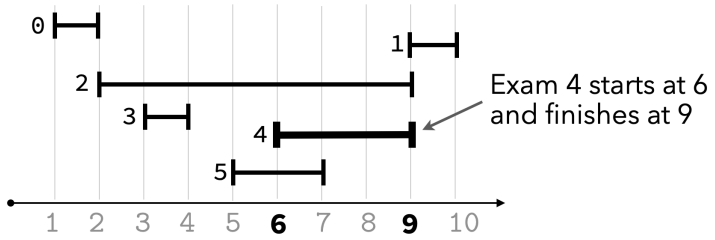
4

26

Question 3 (6 points)

Implement function `int min_overlap(int start[], int finish[], int n)`, which returns the index of the exam with the minimum number of overlaps with other exams. Each exam is represented using an entry in the arrays `start[]` and `finish[]`, where `start[i]` is the start time of exam `i` and `finish[i]` is the finish time of exam `i`.

In the example below, the exams with the minimum number of overlaps are exams **0** and **1** (0 overlaps). You can report any of them.



Exam **0**: No overlaps
Exam **1**: No overlaps
Exam **2**: Overlaps with exams 3, 4 and 5
Exam **3**: Overlaps with exam 2
Exam **4**: Overlaps with exams 2 and 5
Exam **5**: Overlaps with exams 2 and 4

	0	1	2	3	**4**	5
start[] =	1	9	2	3	**6**	5
finish[] =	2	10	9	4	**9**	7

Provide your answer in the following page.

```
// SOLUTION

int min_overlap(int s[], int f[], int n) {
    int min = -1;
    int min_overlaps = 0;

    for (int i = 0; i < n; i++) {
        int overlaps = 0;
        for (int j = 0; j < n; j++) {
            if (f[i] <= s[j] || s[i] >= f[j])
                overlaps++;
        }

        if (min == -1 || overlaps < min_overlaps) {
            min = i;
            min_overlaps = overlaps;
        }
    }

    return min;
}
```