



The King Hussein School for Computing Sciences
Department of Computer Science
11103 - Structured Programming - Spring 2024

Final Exam

Full Name:

Student ID:

Circle your section:

Question	Points	Score
1	5	
2	10	
3	6	
4	4	
5	6	
6	9	
Total	40	

- Dr. Rawan Ghnemat (SuTuTh 9-10)
- Dr. Rawan Ghnemat (SuTuTh 12-13)
- Dr. Rawan Ghnemat (MoWe 8-9½)
- Dr. Samer Sawalha (SuTuTh 8-9)
- Dr. Samer Sawalha (SuTuTh 10-11)
- Dr. Ahmad Klaib (SuTuTh 11-12)
- Dr. Ahmad Klaib (SuTuTh 14-15)
- Dr. Ahmad Klaib (MoWe 12½-14)
- Mr. Alaa Altarazi (SuTuTh 13-14)
- Mr. Alaa Altarazi (MoWe 9½-11)
- Mr. Alaa Altarazi (MoWe 11-12½)
- Dr. Osama Alhaj Hasan (MoWe 11-12½)
- Dr. Mohammad Abu Snober (SuTuTh 11-12)
- Dr. Abdullah Aref (MoWe 14-15½)
- Dr. Amer Al-Badarneh (SuTuTh 12-13)

Saturday 1/6/2024

Question 1. Basic Code Reading and Debugging

(5 marks)

A. [2 points] Find the output of each row independently. Use **ERROR** if the code leads to a compilation or a runtime error. Assume that `x`, `y` and `a` are defined as: `int x = 1, y = 2, a[3] = {9, 2, 5};`

	CODE	OUTPUT
1	<pre>double w = 1 / 2; printf("%f", w);</pre>	
2	<pre>for (int i = 0; i < 3; i++) { printf("%d ", a[i]); break; continue; }</pre>	
3	<pre>for (int i = 0; i < 3; i++) { printf("%d ", a[i]); continue; break; }</pre>	
4	<pre>printf("%d", *a);</pre>	
5	<pre>printf("%d", a[a[1]]);</pre>	
6	<pre>if (y < x < a[0]) printf("YES"); else printf("NO");</pre>	
7	<pre>int w = 5; void f() { int w = 3; } int main() { f(); printf("%d", w); }</pre>	
8	<pre>void f(int* w) { w++; } int main() { f(&x); printf("%d", x); }</pre>	
9	<pre>int f(int w) { return w+1; } int main() { int w = 5; f(w); printf("%d", w); }</pre>	
10	<pre>int f(int w) { return w+1; } int main() { printf("%d", f(f(x))); }</pre>	

Question 2. Basic Code Writing

(10 marks)

- A. [1 point] Complete the function by writing a single line of code that *prints* the result of the equation:

$$\frac{3y - 3}{7x} + x^3$$

```
void compute(int x, int y) {
```

```
}
```

- B. [1 point] Rewrite the following **if**-statement of code without using an **if**-statement.

```
if (x > 7) return 1;
else      return 0;
```

- C. [2 point] Rewrite the following using only a **single if** statement.

```
if (x == 1)
    return 0;
if (y == 2)
    if (x == 3)
        return 0;
return 1;
```

- D. [3 point] Write a piece of code that prints the following shape (assuming *n* is defined and accessible):

```
0 0 0 0 0 0
1 1 1 1 1 1
2 2 2 2 2 2
3 3 3 3 3 3
...
n n n n n n
```

- E. [3 point] Write a piece of code that swaps the first row in an array `a[N][M]` with the last row. Assume that *N* and *M* are defined and accessible.

Question 3. Code Tracing and Comprehension

(6 Marks)

A. Show the output of the code on the right in each of the following cases:

1. If $N = 3$ and $a[] = \{1, 2, 3\}$

2. If $N = 3$ and $a[] = \{3, 2, 1\}$

3. If $N = 1000$ and $a[] = \{1000, 1, 2, 3, \dots, 998, 999\}$

4. If $N = 1000$ and $a[] = \{1000, 999, 998, \dots, 3, 2, 1\}$

```
for (int i = 1; i < N; i++)
    if (a[i] < a[0]) {
        int temp = a[i];
        a[i] = a[0];
        a[0] = temp;
    }
print("%d %d %d", a[0], a[1], a[2]);
```

B. What is the output of the following piece of code?

```
int x = 10;
for (int i = 1; i < x; x--) {
    printf("%d ", i++);
}
```

C. What is the output of the following piece of code?

```
int sum = 0;
for (int i = 1; i <= 3; i++) {
    for (int j = i; j <= 3; j++)
        sum += j;
    printf("%d ", sum);
}
```

Question 4. Recursion

(4 marks)

1. [2 marks] Complete the following *recursive* function, which bubble-sorts the array `a[]`.
(You can pick any number of parameters and use them in your implementation.)

```
void bubbleSort(int arr[], ) {  
  
     base case  
  
    for (  ) {  
        if (arr[i] > arr[i + 1]) {  
            int temp = arr[i];  
            arr[i] = arr[i + 1];  
            arr[i + 1] = temp;  
        }  
    }  
  
     recursive call  
  
}
```

2. [2 marks] What is the output of the following function if called with `n=3` and `a=1`?

```
void mystery(int n, int a) {  
    if (n <= 0) {  
        printf("%d\n", a);  
        return;  
    }  
  
    mystery(n-1, a+1);  
    mystery(n-2, a+2);  
}
```

Question 5. Strings and Pointers

(6 marks)

Implement function `void remove_prefix(char* str, int n)`, which removes the first `n` characters of the string `str`. If `n` is larger than the size of the string, the function removes all the characters in `str`. If `n` is ≤ 0 , the function does nothing.

Examples.	<code>str</code> before	<code>n</code>	<code>str</code> after
	"ABCDEF"	2	"CDEF"
	"ABCDEF"	10	""
	"ABCDEF"	0	"ABCDEF"

Notes.

- You are not allowed to use the array `[]` notation. You must use pointer arithmetic only.
- You are not allowed to use the `string.h` library.

Question 6. 2D Arrays

(9 marks)

Implement function `int HasZeroSquare(int a[H][W], int N, int M)`, which receives a 2D array of size $H \times W$ (where H and W are defined globally) and a size $N \times M$. The function returns `1` if the array has a subarray of size $N \times M$ that is made of all zeroes. The function returns `0` otherwise.

Examples.

The function returns `1` if called with the following array of size $H=6 \times W=5$ and the following values for N and M (for example).

$N=3 \times M=3$	$N=2 \times M=4$	$N=4 \times M=2$	$N=2 \times M=2$																																																																																																																								
<table border="1"><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr><tr><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>2</td></tr><tr><td>1</td><td>0</td><td>0</td><td>2</td><td>2</td></tr></table>	1	2	3	4	5	6	7	8	9	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	2	1	0	0	2	2	<table border="1"><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr><tr><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>2</td></tr><tr><td>1</td><td>0</td><td>0</td><td>2</td><td>2</td></tr></table>	1	2	3	4	5	6	7	8	9	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	2	1	0	0	2	2	<table border="1"><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr><tr><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>2</td></tr><tr><td>1</td><td>0</td><td>0</td><td>2</td><td>2</td></tr></table>	1	2	3	4	5	6	7	8	9	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	2	1	0	0	2	2	<table border="1"><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr><tr><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>2</td></tr><tr><td>1</td><td>0</td><td>0</td><td>2</td><td>2</td></tr></table>	1	2	3	4	5	6	7	8	9	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	2	1	0	0	2	2
1	2	3	4	5																																																																																																																							
6	7	8	9	0																																																																																																																							
1	0	0	0	0																																																																																																																							
1	0	0	0	0																																																																																																																							
1	0	0	0	2																																																																																																																							
1	0	0	2	2																																																																																																																							
1	2	3	4	5																																																																																																																							
6	7	8	9	0																																																																																																																							
1	0	0	0	0																																																																																																																							
1	0	0	0	0																																																																																																																							
1	0	0	0	2																																																																																																																							
1	0	0	2	2																																																																																																																							
1	2	3	4	5																																																																																																																							
6	7	8	9	0																																																																																																																							
1	0	0	0	0																																																																																																																							
1	0	0	0	0																																																																																																																							
1	0	0	0	2																																																																																																																							
1	0	0	2	2																																																																																																																							
1	2	3	4	5																																																																																																																							
6	7	8	9	0																																																																																																																							
1	0	0	0	0																																																																																																																							
1	0	0	0	0																																																																																																																							
1	0	0	0	2																																																																																																																							
1	0	0	2	2																																																																																																																							

The function returns `0` if called on the same array but with values for $N \times M$ like: 5×5 , 1×5 , 5×1 , 6×5 , 5×4 , etc.

NOTE. You are allowed to define and use other functions in your solution if this helps you.

This page is intentionally left blank