(/)

Curriculum
SE Foundations
Average: **108.76%**

Evaluation quiz correction

Evaluation Quiz: Evaluation #1

Date: 2023-03-24

Status: Done

Duration: 18 minutes

Score: 60.0%

# "I don't know": 0

# Success: 9

# Fail: 6

Responses

0. What is wrong with the following code?

```
int n = 5;
int array[5];
int i = 3;

array[n] = i;
```

Score: 0.0

- [✓] **Nothing is wrong**
- [ ] It is impossible to declare the variable `array` this way
- [ ] The array `array` is not entirely initialized
- [ ] **While it is possible to access `array[n]`, we are not supposed to as this is not the array ar**
- [ ] I don't know

Help

(/)

1. What does the macro `TABLESIZE` expand to?

```
#define BUFSIZE 1020
#define TABLESIZE BUFSIZE
#undef BUFSIZE
#define BUFSIZE 37
```

Score: 0.0

- [✓] **1020**
- [ ] **37**
- [ ] nothing
- [ ] I don't know

2. What is the result of `12 % 3`?

Score: 1.0

- [✓] **0**
- [ ] 1
- [ ] 2
- [ ] 3
- [ ] 4
- [ ] I don't know

3. What command(s) can be used to list the symbols stored in a static library?

Select all valid answers

Score: 0.0

- [✓] **`nm`**
- [ ] `ranlib`
- [ ] **`ar`**
- [ ] `ld`
- [ ] I don't know

(/)

4. What is the size of a pointer to an `int` (on a 64-bit architecture)

Score: 0.0

- ☐ 1 byte
- ☐ 2 bytes
- ✓ **4 bytes**
- ☐ **8 bytes**
- ☐ I don't know

5. Are there any memory leaks with the following code (on a 64-bit architecture)?

(/)

```
#include <stdio.h>
#include <stdlib.h>

/**
 * struct list_s - singly linked list
 * @str: string - (malloc'ed string)
 * @len: length of the string
 * @next: points to the next node
 *
 * Description: singly linked list node structure
 * for your project
 */
typedef struct list_s
{
        char *str;
        unsigned int len;
        struct list_s *next;
} list_t;

int main(void)
{
        list_t *node = NULL;
        node = malloc(sizeof(list_t));

        node->len = 3;

        node->str = malloc(sizeof(char) * node->len);
        node->str[0] = 'H';
        node->str[1] = 'i';
        node->str[2] = '\0';

        node->next = NULL;

        free(node);

        return (0);
}
```

Score: 0.0

- ☐ **Yes, 3 bytes of memory were lost**
- [✓] **No, no memory leaks were possible**
- ☐ Yes, 24 bytes of memory were lost
- ☐ Yes, 15 bytes of memory were lost
- ☐ I don't know

6. How many bytes will this statement allocate on a 64-bit machine?

(/)

`malloc(sizeof(char) * 10)`

Score: 1.0

- [✓ **10**]
- 20
- 40
- 80
- I don't know

7. What does this code print?

```
void print(int nb)
{
    printf("%d", nb);
    -- nb;
    if (nb > 0)
    {
        print(nb);
    }
}

int main(void)
{
    print(4);
    return (0);
}
```

Score: 1.0

- [✓ **4321**]
- 43210
- 321
- 3210
- I don't know

8. This `void (*anjula[])(int, float)` is:

Score: 1.0

- ☐ A pointer to a function that takes an `int` and a `float` as parameters and returns nothing

(/)

- ☐ A pointer to a function that takes an array of `int` and `float` as a parameter and returns nothing
- ☐ A pointer to a function that takes an `int` and a `float` as parameters and returns an empty array
- ☑ **An array of pointers to functions that take an `int` and a `float` as parameters and returns nothing**
- ☐ A pointer to an array of functions that take an `int` and a `float` as parameters and returns nothing
- ☐ I don't know

9. How many bytes will this statement allocate on a 64-bit machine?

```
malloc(sizeof(int) * 4)
```

Score: 1.0

- ☐ 4
- ☐ 8
- ☑ **16**
- ☐ 32
- ☐ I don't know

10. What is the value of `n` after the following code is executed?

```
int n = 98;
int *p = &n;

*p++;
```

Score: 1.0

- ☐ 0
- ☑ **98**
- ☐ 99
- ☐ 402
- ☐ I don't know

11. What is the size of `*p` in this code on a 64-bit machine?

```
int **p;
```

Score: 1.0

- ☐ 4 bytes
- ✓ **8 bytes**
- ☐ 16 bytes
- ☐ I don't know

12. The memory space reserved when calling `malloc` is on:

Score: 1.0

- ✓ **The heap**
- ☐ The stack
- ☐ I don't know

13. Given this code:

```
struct point {
   int x;
   int y;
};
struct point my_point = { 3, 7 };
struct point *p = &my_point;
```

To set the member `y` of my variable `my_point` to `98`, I can do (select all valid answers):

Score: 1.0

- ✓ **`my_point.y = 98;`**
- ☐ `my_point->y = 98;`
- ☐ `p.y = 98;`
- ✓ **`(*p).y = 98;`**
- ✓ **`p->y = 98;`**
- ☐ I don't know

14. (/) How much space would you need to allocate for a list node with the following structure on a 64-bit machine?

```
/**
 * struct list_s - singly linked list
 * @str: string - (malloc'ed string)
 * @len: length of the string
 * @next: points to the next node
 *
 * Description: singly linked list node structure
 * for your project
 */
typedef struct list_s
{
    char *str;
    unsigned int len;
    struct list_s *next;
} list_t;
```

Score: 0.0

- ☐ **20 bytes**
- ☐ It's impossible to know without knowing what `str` is
- ✓ **24 bytes**
- ☐ 32 bytes
- ☐ I don't know