Evaluation quiz correction

Evaluation Quiz: Evaluation #1

Date: 2022-04-14

Status: Done

Duration: 36 minutes (including 18 seconds not on the page)

Score: 56.67%

 # "I don't know": 1

 # Success: 8

 # Fail: 6

Responses

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

Score: 1.0

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

- ☐ 1020
- ☑ **37**
- ☐ nothing
- ☐ I don't know

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

Score: 0.0

```
/**
 * 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;
```

- [ ] 20 bytes
- [x] It's impossible to know without knowing what `str` is
- [ ] 24 bytes
- [ ] 32 bytes
- [ ] I don't know

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

Score: 1.0

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


*p++;
```

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

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

Score: 0.0

- ☐ 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

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

Score: 0.0

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

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

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

Score: 1.0

- [ ] 1 byte
- [ ] 2 bytes
- [ ] 4 bytes
- [x] **8 bytes**
- [ ] I don't know

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

Score: 1.0

```
malloc(sizeof(char) * 10)
```

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

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

Score: 1.0

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

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

Score: 0.0

Select all valid answers

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

9. What is wrong with the following code?

Score: 1.0

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


array[n] = i;
```

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

10. What does this code print?

Score: 0.0

```
void print(int nb)
{
    printf("%d", nb);
```

```
    -- nb;
    if (nb > 0)
    {
        print(nb);
    }
}

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

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

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

Score: 0.5

```
#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);
}
```

- 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

- [x] I don't know

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

Score: 1.0

- [x] **The heap**
- [ ] The stack
- [ ] I don't know

13. Given this code:

Score: 0.0

```
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):

- [ ] `my_point.y = 98;`
- [x] `my_point->y = 98;`
- [x] `p.y = 98;`
- [ ] `(*p).y = 98;`
- [ ] `p->y = 98;`
- [ ] I don't know

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

Score: 1.0

```
int **p;
```

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