שאלה 2.1:

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdbool.h>

typedef struct node_t {
    int x;
    struct node_t *next;
} *Node;

typedef enum {
    SUCCESS=0,
    MEMORY_ERROR,
    UNSORTED_LIST,
    NULL_ARGUMENT,
} ErrorCode;

int getListLength(Node list);
bool isListSorted(Node list);
Node mergeSortedLists(Node list1, Node list2, ErrorCode* error_code);

Node createNode(int value);
void destroyList(Node list);
bool memoryAllocationError (Node node, Node list, ErrorCode* error_code);

// creates new node with x equals value given
Node createNode(int value){
    Node node = malloc(sizeof(*node));
    if (!node){ //memory allocation error
        return NULL;
    }
    node->x = value;
    node->next = NULL;
    return node;
}

// destroys the list completely.
void destroyList(Node list){
    Node to_destroy = NULL;
    while(list)
    {
        to_destroy = list;
        list = list->next;
        free(to_destroy);
    }
}
```

```c
//checks if there is memory allocation error in the given node.
//in case of memory allocation error, return true, destroy the list completely
, and puts in variable error_code MEMORY_ERROR.
bool memoryAllocationError (Node node, Node list, ErrorCode* error_code){
    if (!node){
        if(error_code){
            *error_code = MEMORY_ERROR;
        }
        destroyList(list);
        return true;
    }
    return false;
}

Node mergeSortedLists(Node list1, Node list2, ErrorCode* error_code){
    if (getListLength(list1) == 0 || getListLength(list2) == 0){ //unsorted li
st
        if(error_code){
            *error_code = NULL_ARGUMENT;
        }
        return NULL;
    }
    if (isListSorted(list1) == 0 || isListSorted(list2) == 0){ //empty list
        if(error_code){
            *error_code = UNSORTED_LIST;
        }
        return NULL;
    }

    Node ptr_first_list = list1;
    Node ptr_second_list = list2;
    Node merged_list = NULL;
    if (ptr_first_list->x < ptr_second_list->x) {
        merged_list = createNode(ptr_first_list->x);
        ptr_first_list = ptr_first_list->next;
    } else {
        merged_list = createNode(ptr_second_list->x);
        ptr_second_list = ptr_second_list->next;
    }
    if (!merged_list){ //memory allocation error
        return NULL;
    }
    Node iterator = merged_list;
    while(ptr_first_list && ptr_second_list){ //there are more nodes on both l
ists - list 1 and list 2
        if (ptr_first_list->x < ptr_second_list->x){
            iterator->next = createNode(ptr_first_list->x);
            if (memoryAllocationError(iterator-
>next, merged_list, error_code)){ //memory allocation error
                return NULL;
            }
            ptr_first_list = ptr_first_list->next;
            iterator = iterator->next;
        } else{
            iterator->next = createNode(ptr_second_list->x);
            if (memoryAllocationError(iterator-
>next, merged_list, error_code)){ //memory allocation error
                return NULL;
            }
            ptr_second_list = ptr_second_list->next;
            iterator = iterator->next;
        }
    }
    while (ptr_first_list){ //there are more nodes on list 1
        iterator->next = createNode(ptr_first_list->x);
        if (memoryAllocationError(iterator-
>next, merged_list, error_code)){ //memory allocation error
            return NULL;
        }
        ptr_first_list = ptr_first_list->next;
        iterator = iterator->next;
    }
    while (ptr_second_list){ //there are more nodes on list 2
        iterator->next = createNode(ptr_second_list->x);
        if (memoryAllocationError(iterator-
>next, merged_list, error_code)){ //memory allocation error
            return NULL;
        }
        ptr_second_list = ptr_second_list->next;
        iterator = iterator->next;
    }
    if(error_code){
        *error_code = SUCCESS;
    }
    return merged_list;
}
```

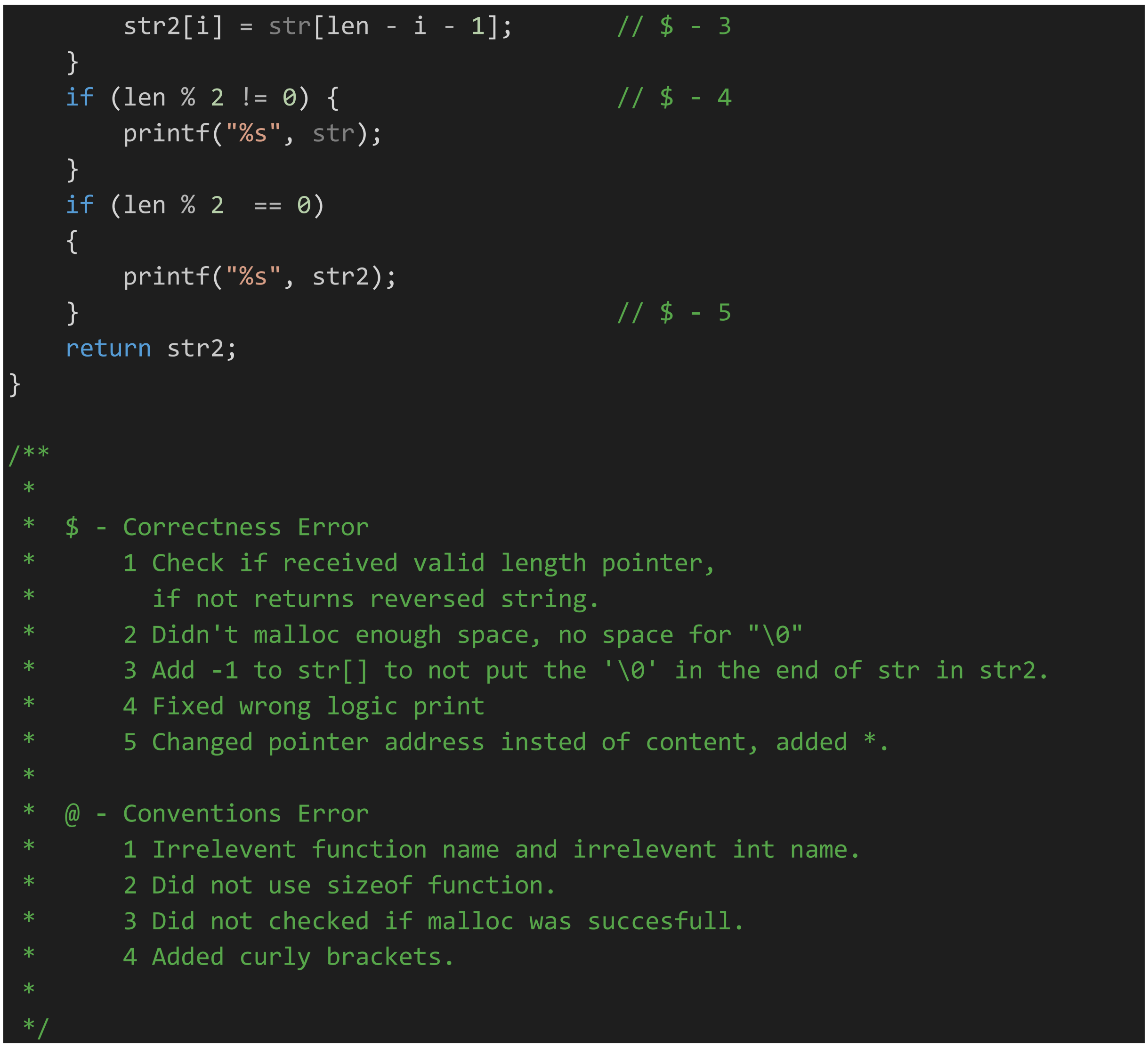
שאלה 2.2:

הקוד המקורי

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

char* foo(char* str, int* x) {
    char* str2;
    int i;
    x = strlen(str);
    str2 = malloc(*x);
    for (i = 0; i < *x; i++)
        str2[i] = str[*x - i];
    if (*x % 2 == 0) {
        printf("%s", str);
    }
    if (*x % 2 != 0)
    {
        printf("%s", str2);
    }
    return str2;
}
```

הקוד המתוקן

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

// $ - Correctness Error
// @ - Conventions Error

char* returnReversedStringAndPrint(char* str,
    int* length) {                        // @ - 1
    int len = strlen(str);
     if(!length) {                        // $ - 1
        *length = len;
    }
    char* str2;
    int i;
    str2 = malloc(sizeof(char)*(len+1));  // @ - 2, $ - 2
    if(!str2)                             // @ - 3
    {
        return NULL;
    }
    for (i = 0; i < len; i++)
    {                                     // @ - 4
```



```
        str2[i] = str[len - i - 1];       // $ - 3
    }
    if (len % 2 != 0) {                    // $ - 4
        printf("%s", str);
    }
    if (len % 2  == 0)
    {
        printf("%s", str2);
    }                                      // $ - 5
    return str2;
}

/**
 *
 *  $ - Correctness Error
 *      1 Check if received valid length pointer,
 *        if not returns reversed string.
 *      2 Didn't malloc enough space, no space for "\0"
 *      3 Add -1 to str[] to not put the '\0' in the end of str in str2.
 *      4 Fixed wrong logic print
 *      5 Changed pointer address insted of content, added *.
 *
 *  @ - Conventions Error
 *      1 Irrelevent function name and irrelevent int name.
 *      2 Did not use sizeof function.
 *      3 Did not checked if malloc was succesfull.
 *      4 Added curly brackets.
 *
 */
```