

Ans_Sheet_1

1. Write the implementation level of the array-based stack we discussed in the lecture in a

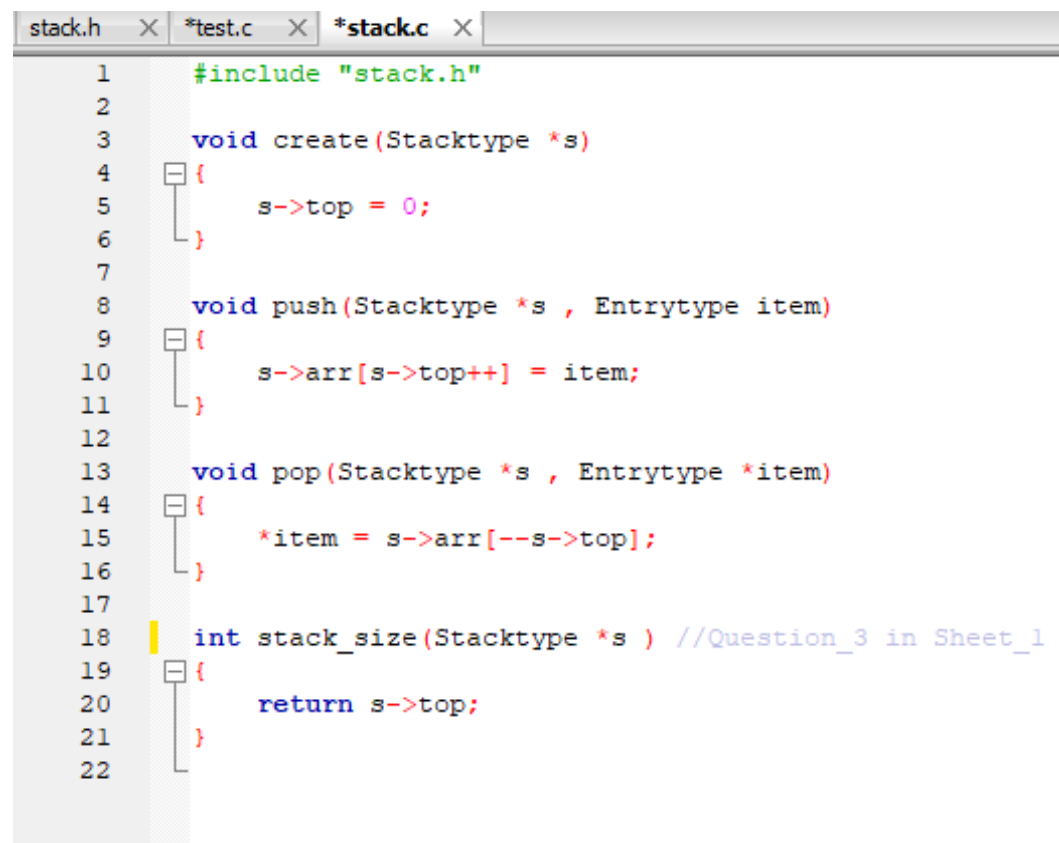
separate file and name it stack.c. Compose the header file stack.h that consists of the prototypes of the functions along with the definition of the stack element type stack

Define stack to be a character. Will you need to include stack.h in stack.c? Why? Compile

stack.c and make sure that the compiler generated the object code file stack.obj.

Ans

Stack.c



```
1      #include "stack.h"
2
3      void create(Stacktype *s)
4      {
5          s->top = 0;
6      }
7
8      void push(Stacktype *s , Entrytype item)
9      {
10         s->arr[s->top++] = item;
11     }
12
13     void pop(Stacktype *s , Entrytype *item)
14     {
15         *item = s->arr[--s->top];
16     }
17
18     int stack_size(Stacktype *s ) //Question_3 in Sheet_1
19     {
20         return s->top;
21     }
22
```

Stack.h

```
stack.h X *test.c X *stack.c X
1  #ifndef STACK_H_INCLUDED
2  #define STACK_H_INCLUDED
3  #define size 10
4  typedef char Entrytype;
5
6  typedef struct
7  {
8      int top;
9      Entrytype arr[size];
10 } Stacktype;
11
12 void create(Stacktype *);
13 void push(Stacktype *, Entrytype);
14 void pop(Stacktype *, Entrytype *);
15 int stack_size (Stacktype *);
16
17
18 #endif // STACK_H_INCLUDED
19
```

2. Write a test program, test.c that calls the stack functions. The program should display and

carry out the appropriate actions based on the following menu:

- (a) Read an element then Push it.
- (b) Pop an element then display it.
- (c) Exit.

At this point, will you be able to build test.c? Why? Build your project; this is the step in which

the "Linker" links the functions from stack.obj to test.obj to create one executable file test.exe.

Run your program.

ANS

Test.c →

في هذا السؤال طلب تسمية فايل Test.c ب main.c

```
stack.h  X  *test.c  X  *stack.c  X
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include "stack.h"
4
5  void print(Stacktype s) //Question_4 in Sheet_1
6  {
7      int i;
8      for(i = 0 ; i < size ; i++)
9      {
10         Entrytype x;
11         pop(&s, &x);
12         printf("%c\n", x);
13     }
14 }
15 //-----|
16 int main()
17 {
18     Stacktype s;
19     create(&s);
20     Entrytype c, item;
21     printf("Enter Your Elements : ");
22     int i;
23     for(i = 0 ; i < size ; i++)
24     {
25         scanf("%c", &c);
26         push(&s, c);
27         printf("The Item Has Been Pushed Into Stack !\n\n");
28     }
29
30     printf("The Size Of Stack Is : %d\n\n", stack_size(&s));
31
32     // Printing Items Without Changing Stack
33     printf("Items Without Changing Stack :\n");
34
35     print(s);
36     //-----
37     printf("\n\nItems With Changing Stack :\n");
38     for(i = 0 ; i < size ; i++)
39     {
40         pop(&s, &item);
41         //printf("The Item Has Been Popped From Stack !\n");
42         printf("%c\n", item);
43     }
44     return 0;
45 }
```

3. In stack.c, write the function stack that returns the size of the stack. Write the pre- and the pre- and post- conditions for this function. Add this option to the menu of the main program defined in number 2.

ANS

It solved in stack .c

pre- condition: the stack is initialized

```
18 int stack_size(Stacktype *s ) //Question_3 in Sheet_1
19 {
20     return s->top;
21 }
22
```

4. Assume that you purchased the stack library stack.obj, along with its header stack.h, from the author; i.e., you do not know any of the details of stack.c. Write a function to print on the screen the contents of a stack without changing the stack. Why shouldn't this function be written in the implementation level?

ANS

هذه ال function يجب ان تكتب في ال user level

Why shouldn't this function be written in the implementation level?

بسبب انه في السؤال يعتبر اننا لا نعرف شيء عن ال stack.c

```
5 void print(Stacktype s) //Question_4 in Sheet_1
6 {
7     int i;
8     for(i = 0 ; i < size ; i++)
9     {
10         Entrytype x;
11         pop(&s, &x);
12         printf("%c\n", x);
13     }
14 }
15 //-----|
```