Sample exercises on Stack (using Array)

You can use the following code to answer the questions given below:

```
#include<stdio.h>
int stack[10],choice,n,top,x,i; // Declaration of variables

void push();
void pop();
void display();

int main()
{
 top = -1;     // Initially there is no element in stack
 printf("\n Enter the size of STACK : ");
 scanf("%d",&n);
 printf("\nSTACK IMPLEMENTATION USING ARRAYS\n");
 do
 {
 printf("\n1.PUSH\n2.POP\n3.DISPLAY\n4.EXIT\n");
 printf("\nEnter the choice : ");
 scanf("%d",&choice);
 switch(choice)
 {
 case 1:
 {
 push();
 break;
 }
 case 2:
```

```
 {
 pop();
 break;
 }
 case 3:
 {
 break;
 }
 default:
 {
 printf ("\nInvalid Choice\n");
 }}}
 while(choice!=3);
 return 0;
}

void push()
{
 if(top >= n - 1)
 {
 printf("\nSTACK OVERFLOW\n");

 }
 else
 {
 printf("Enter a value to be pushed : ");
 scanf("%d",&x);
 top++;            // TOP is incremented after an element is pushed
 stack[top] = x;   // The pushed element is made as TOP
 }}

void pop()
{
```

```
if(top <= -1)
{
printf("\nSTACK UNDERFLOW\n");
}
else
{
printf("\nThe popped element is %d",stack[top]);
top--;    //  Decrement TOP after a pop
}}
```

Write programs for the following questions:

1. A stack, S_1, contains some numbers in arbitrary order. Using two other stacks, say S_2 and S_3 for temporary storage, design an algorithm to sort the numbers in S_1 such that the smallest is at the top of S_1 and the largest is at the bottom. Write a C language code for your algorithm.
2. Assume *S* is a *STACK* and it is having n elements. Write C language code to implement the following queries using *PUSH* and *POP* operations
   (a) Delete the mth element of the stack S from the top (m<n) without missing its top m-1 elements.
   (b) Retain only the elements in the odd position of the stack S and pop out all even positioned elements.
3. Let S be a stack that contains n integers. Let k be an integer given as input. Write a C program to output a pair of numbers in S whose sum is equal to k.
4. Let S_1, S_2 and S_3 be three stacks with $|S_1| = |S_2| = |S_3| = n$ (i.e) all of them wi have same capacity. Assume that only two of the stacks are filled to the capacity with arbitrary non-negative

integers and one of them is empty. Re-arrange the elements in the stack so that even integers are placed in one stack and odd integers in another stack. In case the number of even elements exceeds the number of odd elements, then the extra odd elements that do not fit onto one stack can be placed on the third. The case where odd elements are greater than even is analogous.