

1. write a c program to reverse a string without using additional library functions?

```
#include <stdio.h>

#include <string.h>

void reverseString(char* str) {

    int len = 0;

    while (str[len] != '\0') {

        len++;

    }

    int start = 0;

    int end = len - 1;

    while (start < end) {

        char temp = str[start];

        str[start] = str[end];

        str[end] = temp;

        start++;

        end--;

    }

}

int main() {

    char str[100];

    printf("Enter a string: ");

    fgets(str, sizeof(str), stdin);

    if (str[strlen(str) - 1] == '\n') {
```

```

        str[strlen(str) - 1] = '\0';
    }

reverseString(str);

printf("Reversed string: %s\n", str);

return 0;
}

```

The screenshot shows the OnlineGDB online C compiler interface. The left sidebar contains navigation links: IDE, My Projects, Classroom, Learn Programming, Programming Questions, Jobs, Sign Up, and Login. The main editor area displays a C program for reversing a string. The program includes `<stdio.h>` and `<string.h>`, defines a `reverseString` function, and a `main` function that prompts the user for a string and prints its reverse. The output console shows the input "wipro training" and the reversed output "gniniart orpiw". The program finished with exit code 0.

```

1 #include <stdio.h>
2 #include <string.h>
3 void reverseString(char* str) {
4     int len = 0;
5     while (str[len] != '\0') {
6         len++;
7     }
8     int start = 0;
9     int end = len - 1;
10    while (start < end) {
11        char temp = str[start];
12        str[start] = str[end];
13        str[end] = temp;
14        start++;
15        end--;
16    }
17 }
18 int main() {
19     char str[100];
20     printf("Enter a string: ");
21     fgets(str, sizeof(str), stdin);
22     if (str[strlen(str) - 1] == '\n') {
23         str[strlen(str) - 1] = '\0';
24     }
25     reverseString(str);
26     printf("Reversed string: %s\n", str);
27     return 0;
28 }

```

Input: Enter a string: wipro training
Reversed string: gniniart orpiw
...Program finished with exit code 0
Press ENTER to exit console

2.C program for implementation of selection sort

```

#include <stdio.h>

void swap(int *xp, int *yp)
{
    int temp = *xp;

    *xp = *yp;

    *yp = temp;
}

void selectionSort(int arr[], int n)

```

```

{
    int i, j, min_idx;

    for (i = 0; i < n-1; i++)
    {
        min_idx = i;

        for (j = i+1; j < n; j++)

            if (arr[j] < arr[min_idx])

                min_idx = j;

        if(min_idx != i)

            swap(&arr[min_idx], &arr[i]);

    }
}

void printArray(int arr[], int size)
{
    int i;

    for (i=0; i < size; i++)

        printf("%d ", arr[i]);

    printf("\n");
}

int main()
{
    int arr[] = {64, 25, 12, 22, 11};

    int n = sizeof(arr)/sizeof(arr[0]);

    selectionSort(arr, n);

    printf("Sorted array: \n");
}

```

```

printArray(arr, n);

return 0;

}

```

The screenshot shows the OnlineGDB IDE with a C program for selection sort. The code is as follows:

```

19 {
20     int i, j, min_idx;
21     for (i = 0; i < n-1; i++)
22     {
23         min_idx = i;
24         for (j = i+1; j < n; j++)
25             if (arr[j] < arr[min_idx])
26                 min_idx = j;
27         if (min_idx != i)
28             swap(&arr[min_idx], &arr[i]);
29     }
30 }
31 void printArray(int arr[], int size)
32 {
33     int i;
34     for (i = 0; i < size; i++)
35         printf("%d ", arr[i]);
36     printf("\n");
37 }
38 int main()
39 {
40     int arr[] = {89, 45, 15, 92, 10};
41     int n = sizeof(arr)/sizeof(arr[0]);
42     selectionSort(arr, n);
43     printf("Sorted array: \n");
44     printArray(arr, n);
45     return 0;
46 }

```

The output of the program is:

```

Sorted array:
10 15 18 45 80 92

```

The program finished with exit code 0. The console also shows the message: "Press ENTER to exit console."

3.SWAP TWO NUMBERS:-

```

include <stdio.h>

int main() {

    int a, b, temp;

    int *ptr1, *ptr2;

    printf("Enter the value of a and b: ");

    scanf("%d %d", &a, &b);

    printf("\nBefore swapping a = %d and b = %d", a, b);

    ptr1 = &a;

    ptr2 = &b;

    temp = *ptr1;

```

```

*ptr1 = *ptr2;

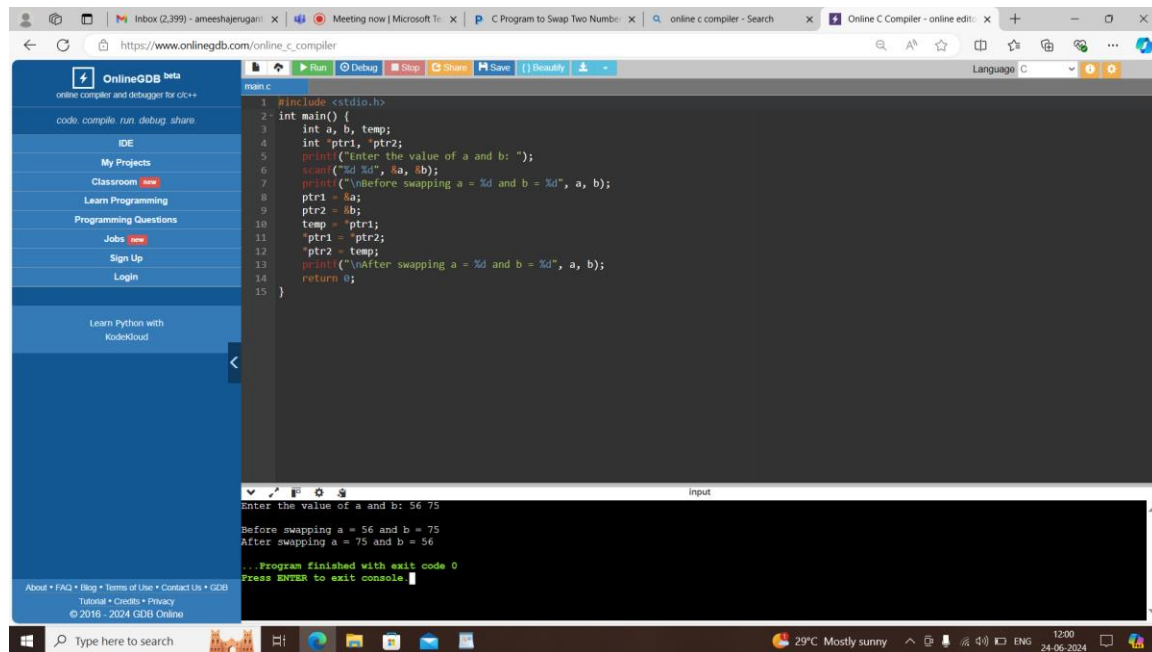
*ptr2 = temp;

printf("\nAfter swapping a = %d and b = %d", a, b);

return 0;

}

```



5. STUDENTS DETAILS :-

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

```
struct student {
```

```
    char name[50];
```

```
    int roll;
```

```
    float marks;
```

```
} s;
```

```
int main() {
```

```
    printf("Enter information:\n");
```

```

printf("Enter name: ");

fgets(s.name, sizeof(s.name), stdin);

printf("Enter roll number: ");

scanf("%d", &s.roll);

printf("Enter marks: ");

scanf("%f", &s.marks);

printf("Displaying Information:\n");

printf("Name: ");

printf("%s", s.name);

printf("Roll number: %d\n", s.roll);

printf("Marks: %.1f\n", s.marks);

return 0;

}

```

The screenshot shows the OnlineGDB website interface. On the left is a sidebar with navigation links like 'IDE', 'My Projects', 'Classroom', 'Learn Programming', 'Programming Questions', 'Jobs', 'Sign Up', 'Login', and 'Learn Python with Kodecloud'. The main area displays a C program with line numbers 1 to 22. The code defines a 'student' struct with fields 'name' (char array), 'roll' (int), and 'marks' (float). The 'main' function prompts the user for name, roll number, and marks, reads the input, and then prints the stored values. Below the code editor is an 'input' field and an 'output' field. The output shows the program's execution: 'Enter roll number: 52', 'Enter marks: 560', 'Displaying Information:', 'Name: arun', 'Roll number: 52', 'Marks: 560.0'. At the bottom, a status bar indicates 'Program finished with exit code 0' and 'Press ENTER to exit console'.