

institute of Computer Technology
B. Tech. Computer Science and Engineering

Sub: ESFP – I
Course Code: 2CSE102
Practical – 16

Name:Dwij vatsal desai

Roll No:BDA-08

Branch:BDA

Class: B

Batch:14

Q.1.Problem Definition:

Mr. Sudhakar Sinha is the puzzle master in a school. He organizes one puzzle session for students and for programming purpose he given two number to the students for variable a and b, Then Mr. Sinha wants to swap value which he given in variable a and b, it means, value of a should be in b and value of b should be in a. So, for the fulfillment of requirement, make a program in c to accept two numbers from user and find out swap value of given input number with the help of pointer.

Code:

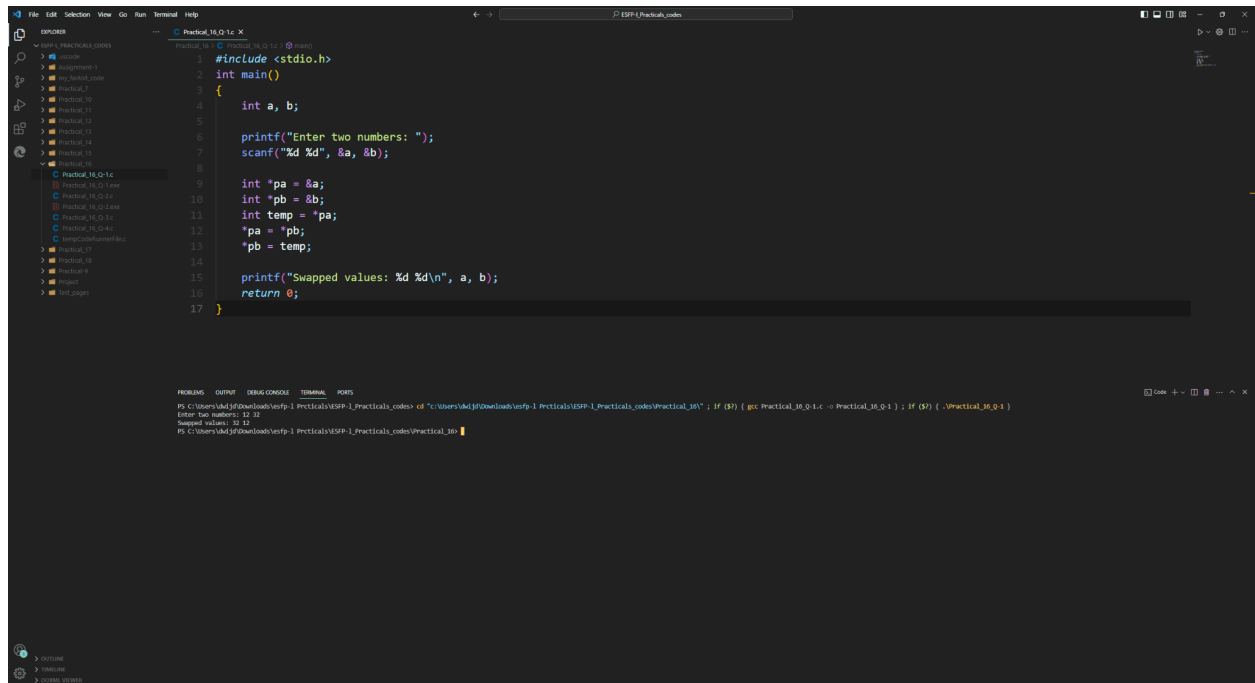
```
#include <stdio.h>
int main()
{
    int a, b;

    printf("Enter two numbers: ");
    scanf("%d %d", &a, &b);

    int *pa = &a;
    int *pb = &b;
    int temp = *pa;
    *pa = *pb;
    *pb = temp;

    printf("Swapped values: %d %d\n", a, b);
    return 0;
}
```

Output:



```
#include <stdio.h>

int main()
{
    int a, b;

    printf("Enter two numbers: ");
    scanf("%d %d", &a, &b);

    int *pa = &a;
    int *pb = &b;
    int temp = *pa;
    *pa = *pb;
    *pb = temp;

    printf("Swapped values: %d %d\n", a, b);
    return 0;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\dwijid\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes> cd "c:\Users\dwijid\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes\Practical_16" ; if (\$?) { g++ Practical_16_0-1.c -o Practical_16_0-1 } ; if (\$?) { .\Practical_16_0-1 }
Enter two numbers: 12 32
Swapped values: 32 12
PS C:\Users\dwijid\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes\Practical_16>

Q.2.Problem Definition:

Description: There is a JAR full of candies for sale at a mall counter. JAR has capacity N, that is JAR can contain maximum N candies when JAR is full. At any point of time. JAR can have M number of Candies where $M \leq N$. Candies are served to the customers. JAR never remained empty as when last k candies are left. JAR is refilled with new candies in such a way that JAR gets full. Write a code to implement the above scenario. Display JAR at counter with available number of candies. The input should be the number of candies one customer can order at a point of time. Update the JAR after each purchase and display JAR at Counter.

Code:

```
#include <stdio.h>

int main()
{
    int N = 10;
    int k = 5;
    int candies = N;
    int *ptrCandies = &candies;
```

```
while (1)
{
    int order;

    printf("Enter the number of candies to order (0 to exit): ");
    scanf("%d", &order);

    if (order < 0 || order > *ptrCandies)
    {
        printf("INVALID INPUT\n");
        printf("NUMBER OF CANDIES LEFT: %d\n", *ptrCandies);
        break;
    }

    printf("NUMBER OF CANDIES SOLD: %d\n", order);

    *ptrCandies -= order;

    if (*ptrCandies < k)
    {
        *ptrCandies = N;
        printf("JAR REFILLED\n");
    }

    printf("NUMBER OF CANDIES AVAILABLE: %d\n", *ptrCandies);
    if (*ptrCandies == 0)
    {
        printf("JAR IS EMPTY\n");
        break;
    }
}
return 0;
}
```

Output:

```
#include <stdio.h>

int ptrCandies = 10;

while (1)
{
    int order;

    printf("Enter the number of candies to order (0 to exit): ");
    scanf("%d", &order);

    if (order < 0 || order > ptrCandies)
    {
        printf("INVALID INPUT\n");
        printf("NUMBER OF CANDIES LEFT: %d\n", ptrCandies);
        break;
    }

    printf("NUMBER OF CANDIES SOLD: %d\n", order);

    ptrCandies -= order;

    if (ptrCandies < k)
    {
        printf("JAR REFILLED\n");
        ptrCandies = 10;
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\dwiid\Downloads\esfp-1 Prcticals\ESFP-1 Practicals_codes\Practical_16> .\Practical_16_02.exe
Enter the number of candies to order (0 to exit): 2 3 4 5 6
NUMBER OF CANDIES SOLD: 2
NUMBER OF CANDIES AVAILABLE: 8
Enter the number of candies to order (0 to exit): NUMBER OF CANDIES SOLD: 3
NUMBER OF CANDIES AVAILABLE: 5
Enter the number of candies to order (0 to exit): NUMBER OF CANDIES SOLD: 4
JAR REFILLED
NUMBER OF CANDIES AVAILABLE: 10
Enter the number of candies to order (0 to exit): NUMBER OF CANDIES SOLD: 5
NUMBER OF CANDIES AVAILABLE: 5
Enter the number of candies to order (0 to exit): INVALID INPUT
NUMBER OF CANDIES LEFT: 5
PS C:\Users\dwiid\Downloads\esfp-1 Prcticals\ESFP-1 Practicals_codes\Practical_16>

```
PS C:\Users\dwiid\Downloads\esfp-1 Prcticals\ESFP-1 Practicals_codes> cd "c:\Users\dwiid\Downloads\esfp-1 Prcticals\ESFP-1 Practicals_codes\Practical_16"
Enter the number of candies to order (0 to exit): 2 3 4 5 6
NUMBER OF CANDIES SOLD: 2
NUMBER OF CANDIES AVAILABLE: 8
Enter the number of candies to order (0 to exit): NUMBER OF CANDIES SOLD: 3
NUMBER OF CANDIES AVAILABLE: 5
Enter the number of candies to order (0 to exit): NUMBER OF CANDIES SOLD: 4
JAR REFILLED
NUMBER OF CANDIES AVAILABLE: 10
Enter the number of candies to order (0 to exit): NUMBER OF CANDIES SOLD: 5
NUMBER OF CANDIES AVAILABLE: 5
Enter the number of candies to order (0 to exit): INVALID INPUT
NUMBER OF CANDIES LEFT: 5
PS C:\Users\dwiid\Downloads\esfp-1 Prcticals\ESFP-1 Practicals_codes\Practical_16>
```

Q.3.Problem Definition:

In a competition, there are three 3 problems: P, Q, R. Alice challenges Bob that problem R will be the most difficult, whereas Bob predicts that problem Q will be the most difficult. You are given three integers SP,SQ,SR, which represent the number of successful submissions of the problems P, Q, R. Each problem is guaranteed to have a varied number of submissions. Decide who will win the challenge.

Code:

```
#include <stdio.h>

int determineWinner(int SP, int SQ, int SR)
```

```

{
    if (SP < SQ && SP < SR)
    {
        return 0;
    }
    else if (SQ < SP && SQ < SR)
    {
        return 1;
    }
    else
    {
        return 2;
    }
}

int main()
{
    int T;

    scanf("%d", &T);

    while (T--)
    {
        int SP, SQ, SR;

        scanf("%d %d %d", &SP, &SQ, &SR);

        int winnerCode = determineWinner(SP, SQ, SR);

        if (winnerCode == 0)
        {
            printf("Draw\n");
        }
        else if (winnerCode == 1)
        {
            printf("Bob\n");
        }
        else
        {
            printf("Alice\n");
        }
    }
    return 0;
}

```

Output:

```
20 int T;  
21  
22 scanf("%d", &T);  
23  
24 while (T--)  
25 {  
26     int SP, SQ, SR;  
27  
28     scanf("%d %d %d", &SP, &SQ, &SR);  
29  
30     int winnerCode = determineWinner(SP, SQ, SR);  
31  
32     if (winnerCode == 0)  
33     {  
34         printf("Draw\n");  
35     }  
36     else if (winnerCode == 1)  
37     {  
38         printf("Bob\n");  
39     }  
40     else
```

```
PS C:\Users\dwijd\Downloads\esfp-1 Prcticals\ESFP-1 Practicals_codes> cd "c:\Users\dwijd\Downloads\esfp-1 Prcticals\ESFP-1 Practicals_codes\Practical_16"  
3  
1 2 3  
Draw  
16 8 10  
Bob  
14 15 9  
Alice  
PS C:\Users\dwijd\Downloads\esfp-1 Prcticals\ESFP-1 Practicals_codes\Practical_16>
```

Q.4.Problem Definition:

Mr. Jostin D'Souza is the admin head in a school, he wants to see all the students' name in alphabet ascending order for easy indexing of roll number. So, that for the fulfillment of the above said requirement make an appropriate program in C to accept 5 student names from user and then display all name in ascending order.

[Note: Solve this program using pointer]

Code:

```

#include <stdio.h>
#include <string.h>

int sortNames(char *names[], int n)
{
    int i, j;
    char *temp;
    for (i = 0; i < n - 1; i++)
    {
        for (j = 0; j < n - i - 1; j++)
        {
            if (strcmp(names[j], names[j + 1]) > 0)
            {
                temp = names[j];
                names[j] = names[j + 1];
                names[j + 1] = temp;
            }
        }
    }
}

int main()
{
    int n;
    printf("Enter Number of student names: ");
    scanf("%d", &n);

    char *studentNames[n];
    printf("Enter %d student names:\n", n);

    for (int i = 0; i < n; i++)
    {
        char name[50];
        scanf("%s", name);
        studentNames[i] = strdup(name);
    }

    sortNames(studentNames, n);
    printf("Names in alphabetical ascending order:\n");
    for (int i = 0; i < n; i++)
    {
        printf("%s\n", studentNames[i]);
        free(studentNames[i]);
    }
}

```

```
}  
    return 0;  
}
```

Output:

```
Output  
Enter Number of student names: 3  
Enter 3 student names:  
re  
yt  
qe  
Names in alphabetical ascending order:  
qe  
re  
yt
```