Assignment 7: Function

Submitted By: U19CS012(D-12)

1. Calculate x^y i.e. pow(x, y) using recursion.

Code:

```c
#include <stdio.h>
#include <math.h>
typedef unsigned long long ull;
// Divide and Conquer Approach a^n = a^(n/2)*a^(n/2)
ull power(ull a, ull n)
{
    if (n == 0)
        return 1;
    if (n == 1)
        return a;
    ull t = power(a, (n / 2));
    return t * t * power(a, n % 2);
}
//For a^3 = (a^1)*(a^1)*(a^(3%2))
//For a^5 = (a^2)*(a^2)*(a^(5%2)) --> recursive call for a^2
//        = ((a*a))*((a*a))*(a^(5%2))
// Method 2 for Small Numbers (Not Effecient)
ull power2(ull base, ull a)
{
    if (a != 0)
        return (base * power(base, a - 1));
    else
        return 1;
}

void main()
{
    ull x, y;
    printf("\nFor Calculating x^y : \n");
    printf("\nEnter a Base Number(x) : ");
    scanf("%llu", &x);
    printf("\nEnter a Exponent Number(y) : ");
    scanf("%llu", &y);
    ull ans = power(x, y);
    printf("\nThe Result of %llu raised to %llu : %llu ", x, y, ans);
    //Built in Function Conflict with Same Name "pow",
    // So changed to "power"
}
```

Output:

```
For Calculating x^y :

Enter a Base Number(x) : 23

Enter a Exponent Number(y) : 10

The Result of 23 raised to 10 : 41426511213649
```

2. Reads an n x n matrix and displays the sum of elements for the main diagonal.

Code:

```c
#include <stdio.h>
const int max = 100;
void main_sum(int mat[][max], int n)
{
    int sum = 0;
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            if (i == j)
                sum += mat[i][j];
        }
    }
    printf("\n The Sum of Main Diagonal : %d", sum);
}

void main()
{
    int n;
    int mat[max][max];
    printf("\nEnter the n for (n*n) matrix (n<100) : ");
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            printf("matrix[%d][%d] = ", i, j);
            scanf("%d", &mat[i][j]);
        }
    }
    printf("\nThe Input Matrix :\n");
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            printf("%d ", mat[i][j]);
```

```
        }
        printf("\n");
    }
    main_sum(mat, n); // Function Call
}
```

Output:

```
Enter the n for (n*n) matrix (n<100) : 4
matrix[0][0] = 10
matrix[0][1] = 11
matrix[0][2] = 12
matrix[0][3] = 13
matrix[1][0] = 14
matrix[1][1] = 15
matrix[1][2] = 16
matrix[1][3] = 17
matrix[2][0] = 18
matrix[2][1] = 19
matrix[2][2] = 20
matrix[2][3] = 21
matrix[3][0] = 22
matrix[3][1] = 23
matrix[3][2] = 24
matrix[3][3] = 25

The Input Matrix :
10 11 12 13
14 15 16 17
18 19 20 21
22 23 24 25

 The Sum of Main Diagonal : 70
```

3. Insert a word in a string at a position given by user.

Code:

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
void insert_str(char str1[100], char str2[100], int p)
```

```
{
    int i;
    int l1 = strlen(str1);
    int l2 = strlen(str2);
    // Convert abc|"efg" --> abc|string length|"efg"
    for (i = p; i < l1; i++)
    {
        str1[i + l2] = str1[i];
    }
    // convert abc|string_length|"efg" --> abc|pqrs|efg
    for (i = 0; i < l2; i++)
    {
        str1[p + i] = str2[i];
    }
    str2[l2 + 1] = '\0'; //null  character at end
    printf("The Final String After inserting is %s", str1);
}

void main()
{
    char str1[100], str2[100];
    int position = 0;
    printf("Enter the string 1\n");
    gets(str1);
    printf("Enter the string 2(to be inserted)\n");
    gets(str2);
    printf("Enter the position where the string is to be inserted\n");
    scanf("%d", &position);
    insert_str(str1,str2,position);
}
```

Output:

```
Enter the string 1
My India
Enter the string 2(to be inserted)
Atmanirbhar
Enter the position where the string is to be inserted
3
The Final String After inserting is My Atmanirbhar India
```

4. Implement a function named as flip; which will take a number as input and flip its last N digits.

For example: flip (123, 2) = 132; (here N=2)

Flip (12345, 3) = 12543 (here N=3)

Code:

```c
#include <stdio.h>
// #include <math.h>
int power(int a, int n)
{
    if (n == 0)
        return 1;
    if (n == 1)
        return a;
    int t = power(a, (n / 2));
    return t * t * power(a, n % 2);
}
int flip(int n, int k)
{
    int edit_part = n % (power(10, k)); //1234
    n = n - edit_part;                  //1000
    int rev_edit = 0, remainder = 0;
    while (edit_part != 0)
    {
        remainder = edit_part % 10;
        rev_edit = rev_edit * 10 + remainder;
        edit_part /= 10;
    }
    n = n + rev_edit;
    return n;
}

void main()
{
    int n, k;
    printf("\nEnter a Number : ");
    scanf("%d", &n);
    printf("\nEnter Number of digits to Flip : ");
    scanf("%d", &k);
    int ans = flip(n, k);
    printf("\nThe Number after Flipping is : %d", ans);
}
```

Output:

Test Case 1:

```
Enter a Number : 123

Enter Number of digits to Flip : 2

The Number after Flipping is : 132
```

Test Case 2:

```
Enter a Number : 12345

Enter Number of digits to Flip : 3

The Number after Flipping is : 12543
```

Test Case 3:

```
Enter a Number : 1234567890

Enter Number of digits to Flip : 5

The Number after Flipping is : 1234509876
```

5. For the flip function in Q-5, verify that flip(flip(N,k), k)= N.

Code:

```c
#include <stdio.h>
// #include <math.h>
int power(int a, int n)
{
    if (n == 0)
        return 1;
    if (n == 1)
        return a;
    int t = power(a, (n / 2));
    return t * t * power(a, n % 2);
}
int flip(int n, int k)
{
    int edit_part = n % (power(10, k)); //1234
    n = n - edit_part;              //1000
    int rev_edit = 0, remainder = 0;
    while (edit_part != 0)
    {
        remainder = edit_part % 10;
        rev_edit = rev_edit * 10 + remainder;
        edit_part /= 10;
    }
    n = n + rev_edit;
    return n;
}

void main()
```

```
{
    int n, k;
    printf("\nEnter a Number : ");
    scanf("%d", &n);
    printf("\nEnter Number of digits to Flip : ");
    scanf("%d", &k);
    int ans = flip(n, k);
    printf("\nThe Number after First Flipping is : %d\n", ans);
    int double_flip = flip(ans,k);
    printf("\nThe Number after Double Flipping is : %d\n", double_flip);
}
```

Output:

```
Enter a Number : 123456789

Enter Number of digits to Flip : 5

The Number after First Flipping is : 123498765

The Number after Double Flipping is : 123456789
```

After Operating Double Flipping, we get the same Number Again.

Hence, Proved that flip (flip (n, k), k) = n

(We can Conclude that Even Double Swapping of Same No. Of Digits leads to Same Number).

Submitted By:

Bhagya Rana

U19CS012 (D-12) (CSE, SVNIT)