CSE ASSIGNMENT - 8

Problem 1

Question:

> *Write a program to get five single digit numbers in an array and form a number using the array elements. (Use function with the array as argument and return the number)*

Code:

```c
// Program to get 5 elements in an array and stitch them together

#include <stdio.h>

int stitch(const int num_arr[], const int size)
{
    int stitched = 0;

    for (int i = 0; i < size; i++)
    {
        stitched = (stitched * 10) + num_arr[i];
    }

    return stitched;
}

int main(int argc, char const *argv[])
{
    int num_arr[5];

    for (int i = 0; i < 5; i++)
    {
        scanf(" %d", num_arr + i);
    }

    printf("The number was : %d", stitch(num_arr, sizeof(num_arr) / 4));

    return 0;
}
```

Output:

```
3
4
3
5
```

```
6
The number was : 34356
```

Problem 2

Question:

> *Give a one-dimensional array with positive integer values, rearrange the array such that the odd numbers precede the even numbers.*

Code:

```c
// Program to re-arrange a positive integer array in such a way that the odd numbers
preceed the even numbers

#include <stdio.h>

void get_int_input(const char string[], int *invar)
{
    printf("%s", string);
    scanf(" %d", invar);
}

void rearrange(int arr[], const int size)
{
    int replace = 0, new_arr[size];

    for (int i = 0; i < size; i++)
    {
        if (arr[i] % 2 != 0)
        {
            new_arr[replace] = arr[i];
            replace++;
        }
    }

    for (int i = 0; i < size; i++)
    {
        if (arr[i] % 2 == 0)
        {
            new_arr[replace] = arr[i];
            replace++;
        }
    }

    printf("The re-arranged array is : { ");

    for (int i = 0; i < size; i++)
    {
        printf("%d, ", *(new_arr + i));
```

```
    }

    printf("\b\b }\n");
}

int main(int argc, char const *argv[])
{
    int n;

    get_int_input("How many elements would you like to have : ", &n);

    int arrarr[n];

    for (int i = 0; i < n; i++)
    {
        get_int_input("", arrarr + i);
    }

    rearrange(arrarr, n);

    return 0;
}
```

Output:

```
How many elements would you like to have : 9
1
3
4235
546
23
54
65
76
34
The re-arranged array is : { 1, 3, 4235, 23, 65, 546, 54, 76, 34 }
```

Problem 3

Question:

> *Given an integer matrix of size m x n with positive and negative values, separate the integers based on the sign and store in two 1D arrays (Positive & Negative arrays). Find the maximum and minimum values in both the arrays. Sample Input: Input matrix: 120 117 136 -110 -150 128 135 114 -149 Sample Output: Array with positive elements: 120 117 136 128 135 114 Maximum = 136 Array with negative elements: -110 -150 -149 Maximum = -110*

Code:

```c
// Program to find the maximum positive and negative number in a given matrix after splitting the terms into two arrays containing positive and negative elements respectively

#include <stdio.h>

int main(int argc, char const *argv[])
{
    int m, n;

    printf("How many rows : ");
    scanf(" %d", &m);

    printf("How many columns : ");
    scanf(" %d", &n);

    int matrix[m][n];

    for (int i = 0; i < m; i++)
    {
        for (int j = 0; j < n; j++)
        {
            printf("Enter the element in %d x %d : ", i + 1, j + 1);
            scanf(" %d", &matrix[i][j]);
        }
    }

    int pos_elems[m * n], neg_elems[m * n], pos_counter = 0, neg_counter = 0, pos_max = 0, neg_max = -2147483647;

    for (int i = 0; i < m; i++)
    {
        for (int j = 0; j < n; j++)
        {
            if (matrix[i][j] > 0)
            {
                pos_elems[pos_counter] = matrix[i][j];
                pos_counter++;

                if (matrix[i][j] > pos_max)
                {
                    pos_max = matrix[i][j];
                }
            }
            else if (matrix[i][j] < 0)
            {
                neg_elems[neg_counter] = matrix[i][j];
                neg_counter++;

                if (matrix[i][j] > neg_max)
                {
                    neg_max = matrix[i][j];
```

```
                }
            }
        }
    }

    printf("The positive array is : { ");

    for (int i = 0; i < pos_counter; i++)
    {
        printf("%d, ", *(pos_elems + i));
    }

    printf("\b\b }\n");

    printf("The negative array is : { ");

    for (int i = 0; i < neg_counter; i++)
    {
        printf("%d, ", *(neg_elems + i));
    }

    printf("\b\b }\n");

    printf("The largest element in the positive array is : %d\n", pos_max);
    printf("The largest element in the negative array is %d\n", neg_max);

    return 0;
}
```

Output:

```
How many rows : 3
How many columns : 3
Enter the element in 1 x 1 : 1
Enter the element in 1 x 2 : 2
Enter the element in 1 x 3 : 3
Enter the element in 2 x 1 : -4
Enter the element in 2 x 2 : -2
Enter the element in 2 x 3 : 1
Enter the element in 3 x 1 : -7
Enter the element in 3 x 2 : 2
Enter the element in 3 x 3 : 3
The positive array is : { 1, 2, 3, 1, 2, 3 }
The negative array is : { -4, -2, -7 }
The largest element in the positive array is : 3
The largest element in the negative array is -2
```

Problem 4

Question:

> *Write a C program to count the number of non-zero elements in a two-dimensional matrix using function with 2D array as argument. Sample Input: Input matrix: 12 179 0 100 0 1 5 4 0 Sample Output: Number of non-zero elements: 6*

Code:

```c
// Program to count the total number of non-zero elements in a matrix

#include <stdio.h>

int main(int argc, char const *argv[])
{
    int m, n;

    printf("How many rows : ");
    scanf(" %d", &m);

    printf("How many columns : ");
    scanf(" %d", &n);

    int temp, counter = 0;

    for (int i = 0; i < m; i++)
    {
        for (int j = 0; j < n; j++)
        {
            printf("Enter the element in %d x %d : ", i + 1, j + 1);
            scanf(" %d", &temp);

            if (temp != 0)
            {
                counter++;
            }
        }
    }

    printf("The total number of non-zero elements in the given matrix is : %d\n",
counter);

    return 0;
}
```

Output:

```
How many rows : 3
How many columns : 3
Enter the element in 1 x 1 : 1
Enter the element in 1 x 2 : 0
Enter the element in 1 x 3 : 1
Enter the element in 2 x 1 : 0
Enter the element in 2 x 2 : 2
```

```
Enter the element in 2 x 3 : 3
Enter the element in 3 x 1 : 0
Enter the element in 3 x 2 : 0
Enter the element in 3 x 3 : 5
The total number of non-zero elements in the given matrix is : 5
```

Problem 5

Question:

> *Write a C program to convert the string “CEASER” to “HJFXJW” without using inbuilt functions.*

Code:

```c
// Program to convert 'CEASER' to 'HJFXJW'

#include <stdio.h>

int main(int argc, char const *argv[])
{
    char ceaser[7] = "CEASER";

    for (int i = 0; i < 6; i++)
    {
        printf("%c", ceaser[i] + 5);
    }

    return 0;
}
```

Output:

```
HJFXJW
```

Problem 6

Question:

> *Given an input string “SASTRA Deemed University”, write a C program to extract the substring with the following inputs: Number of characters to be extracted, the location from where it has to start extraction. Print the extracted substring and its reversed form. (Use user-defined functions and do not use string-inbuilt functions) Sample input: Enter the input string: SASTRA Deemed University Enter the length of the substring to be extracted: 5 Enter the location from where extraction should start: 4 Sample output: The extracted substring: RA De The reverse form of extracted substring: eD AR*

Code:

```c
// Program to extract a substring

#include <stdio.h>

void print_chars(const char str[], int start, int end)
{
    if (start < end)
    {
        for (int i = start; i < end; i++)
        {
            printf("%c", *(str + i));
        }
    }
    else
    {
        for (int i = start - 1; i > end - 1; i--)
        {
            printf("%c", *(str + i));
        }
    }

    printf("\n");
}

int main(int argc, char const *argv[])
{
    char instr[100];
    int start, end;

    printf("Enter a string : ");
    fgets(instr, 100, stdin);

    printf("Initial position of slice : ");
    scanf(" %d", &start);

    printf("Number of characters to slice off : ");
    scanf(" %d", &end);

    end += start;

    print_chars(instr, start, end);
    print_chars(instr, end, start);

    return 0;
}
```

Output:

```
Enter a string : SASTRA Deemed University
Initial position of slice : 4
Number of characters to slice off : 5
RA De
eD AR
```

Problem 7

Question:

> *Given an input string “Stay calm and Keep programming!!”, write a C program to chop this string and store it as an array of strings. Find the lengthiest and shortest string from the array of strings without using inbuilt functions. Sample Input: Stay calm and Keep programming!! Sample Output: Chopped Strings! Stay calm and Keep programming!! ---------- programming is the lengthiest. and is the shortest.*

Code:

```c
// Program to chop up a string based on spaces and also to find the longest and the shortest strings among the chopped strings

#include <stdio.h>

int main(int argc, char const *argv[])
{
    char beegstring[100];

    printf("Enter a string : ");
    fgets(beegstring, 100, stdin);

    char smolstrings[100][100];
    int current_smol = 0, current_smol_char_pos = 0, smol_str_len_arr[100];

    for (int i = 0; i < 100; i++)
    {
        if (beegstring[i] == ' ' || beegstring[i] == '\n')
        {
            smol_str_len_arr[current_smol] = current_smol_char_pos;
            smolstrings[current_smol][current_smol_char_pos] = '\0';
            current_smol_char_pos = 0;

            current_smol++;
            continue;
        }
        else if (beegstring[i] == '\0')
        {
            break;
        }
        else
```

```
        {
            smolstrings[current_smol][current_smol_char_pos] = beegstring[i];
            current_smol_char_pos++;
        }
    }

    int max = 0, min = 2147483647, max_index = 0, min_index = 0;

    for (int i = 0; i < current_smol; i++)
    {
        if (smol_str_len_arr[i] > max)
        {
            max = smol_str_len_arr[i];
            max_index = i;
        }
        if (smol_str_len_arr[i] < min)
        {
            min = smol_str_len_arr[i];
            min_index = i;
        }
    }

    for (int i = 0; i < current_smol; i++)
    {
        printf("%s\n", smolstrings[i]);
    }

    printf("The longest slice is '%s' measuring %d\n", smolstrings[max_index], max);
    printf("The shortest string is '%s' measuring %d", smolstrings[min_index], min);

    return 0;
}
```

Output:

```
Enter a string : Stay calm and Keep programming!!
Stay
calm
and
Keep
programming!!
The longest slice is 'programming!!' measuring 13
The shortest string is 'and' measuring 3
```
