**Module 1 Assignment**

**TOPIC 1:**

#include <stdio.h>

#include <string.h>

typedef struct {

char dayname[10];

int task;

char tasks[3][50];

} calendar;

int main() {

calendar d[7];

strcpy(d[0].dayname, "Monday");

d[0].task = 0;

strcpy(d[1].dayname, "Tuesday");

d[1].task = 0;

strcpy(d[2].dayname, "Wednesday");

d[2].task = 0;

strcpy(d[3].dayname, "Thursday");

d[3].task = 0;

strcpy(d[4].dayname, "Friday");

d[4].task = 0;

strcpy(d[5].dayname, "Saturday");

d[5].task = 0;

strcpy(d[6].dayname, "Sunday");

d[6].task = 0;

int c;

while (1) {

printf("\nMenu:\n1. Enter tasks\n2. Display tasks\n3. Exit\nChoose an option: ");

scanf("%d", &c);

if (c == 1) {

char day[10];

printf("Enter day: ");

scanf("%s", day);

int i = 0;

while (i < 7 && strcmp(d[i].dayname, day) != 0) {

i++;

}

if (i == 7) {

printf("Invalid day entered!\n");

continue;

}

printf("Till now %d tasks entered out of 3\n", d[i].task);

printf("How many new tasks do you want to enter? ");

int j = 0;

scanf("%d", &j);

if (j > 3 - d[i].task) {

printf("Sorry! That exceeds the limit of 3 tasks per day. Try again.\n");

} else {

for (int k = d[i].task; k < j + d[i].task; k++) {

printf("Enter Task %d: ", k + 1);

scanf(" %[^\n]s", d[i].tasks[k]);

}

d[i].task += j;

}

}

else if (c == 2) {

char day[10];

printf("Enter day to display tasks: ");

scanf("%s", day);

int i = 0;

while (i < 7 && strcmp(d[i].dayname, day) != 0) {

i++;

}

if (i == 7) {

printf("Invalid day entered!\n");

continue;

}

printf("\nDetails\nDay: %s\nNumber of tasks entered: %d\nTasks:\n", d[i].dayname, d[i].task);

for (int k = 0; k < d[i].task; k++) {

printf("- %s\n", d[i].tasks[k]);

}

}

else if (c == 3) {

printf("Exiting...\n");

break;

}

else {

printf("Invalid choice. Try again.\n");

}

}

return 0;

}

**Output:**

Menu:

1. Enter tasks

2. Display tasks

3. Exit

Choose an option: 2

Enter day to display tasks: Monday

Details

Day: Monday

Number of tasks entered: 0

Tasks:

Menu:

1. Enter tasks

2. Display tasks

3. Exit

Choose an option: 1

Enter day: Monday

Till now 0 tasks entered out of 3

How many new tasks do you want to enter? 5

Sorry! That exceeds the limit of 3 tasks per day. Try again.

Menu:

1. Enter tasks

2. Display tasks

3. Exit

Choose an option: 1

Enter day: Monday

Till now 0 tasks entered out of 3

How many new tasks do you want to enter? 2

Enter Task 1: Go to Market

Enter Task 2: Study

Menu:

1. Enter tasks

2. Display tasks

3. Exit

Choose an option: 2

Enter day to display tasks: Monday

Details

Day: Monday

Number of tasks entered: 2

Tasks:

- Go to Market

- Study

Menu:

1. Enter tasks

2. Display tasks

3. Exit

Choose an option: 3

Exiting...

Process returned 0 (0x0) execution time : 60.198 s

Press any key to continue.

**TOPIC 2:**

#include <stdio.h>

void reorder(int\* arr, int n) {

int\* end = arr + n;

for (int\* c = arr + 1; c < end; c++) {

if (\*c % 2 == 0) {

int value = \*c;

int\* p = c;

while (p > arr && (\*(p - 1) % 2 != 0)) {

\*p = \*(p - 1);

p--;

}

\*p = value;

}

}

}

int main() {

int n;

printf("Enter the number of array elements : ");

scanf("%d",&n);

int arr[n];

for(int i=0;i<n;i++){

printf("Element %d: ",i+1);

scanf("%d",&arr[i]);

}

reorder(arr, n);

printf("Modified array:\n");

for (int i = 0; i < n; i++) {

printf("%d ", \*(arr + i));

}

return 0;

}

**OUTPUT:**

Enter the number of array elements : 5

Element 1: 1

Element 2: 2

Element 3: 3

Element 4: 4

Element 5: 5

Modified array:

2 4 1 3 5

Process returned 0 (0x0) execution time : 3.046 s

Press any key to continue.

**TOPIC 3:**

#include <stdio.h>

#define n 100

int main() {

int rows, cols;

int matrix[n][n];

printf("Enter number of rows and columns: ");

scanf("%d %d", &rows, &cols);

printf("Enter matrix elements (row-wise, sorted by row and column):\n");

for (int i = 0; i < rows; i++) {

for (int j = 0; j < cols; j++) {

scanf("%d", \*(matrix + i) + j);

}

}

int key;

printf("Enter the key to search: ");

scanf("%d", &key);

int i = 0, j = cols - 1;

int found = 0;

while (i < rows && j >= 0) {

int current = \*(\*(matrix + i) + j);

if (current == key) {

printf("Key %d found at position (%d, %d)\n", key, i, j);

found = 1;

break;

} else if (current > key) {

j--;

} else {

i++;

}

}

if (!found) {

printf("Key %d not found in the matrix.\n", key);

}

return 0;

}

**OUTPUT:**

Enter number of rows and columns: 3

3

Enter matrix elements (row-wise, sorted by row and column):

1 2 3

4 5 6

7 8 9

Enter the key to search: 5

Key 5 found at position (1, 1)

Process returned 0 (0x0) execution time : 17.373 s

Press any key to continue.