Assignment – 09

```c
// Q1. WAP to Store Information of 3 students and Display it Using Structure.

#include <stdio.h>

struct Student
{
    char Name[100];
    char RegNo[10];
    char Branch[100];
};


int main(){
    struct Student s[3];

    for (int i = 0; i < 3; i++)
    {
        printf("Enter name : ");
        scanf("%s", &s[i].Name);

        printf("Enter Registration No. : ");
        scanf("%s", &s[i].RegNo);

        printf("Enter Branch : ");
        scanf("%s", &s[i].Branch);
        printf("\n");
    }

    for (int i = 0; i < 3; i++)
    {
        printf("%s\n%s\n%s\n\n", s[i].Name, s[i].RegNo, s[i].Branch);
    }

    return 0;
}
```

Output :-

```
Enter name : Raj
Enter Registration No. : 2201020210
Enter Branch : CSIT

Enter name : Amit
Enter Registration No. : 2201020211
Enter Branch : CSE

Enter name : Itzu
Enter Registration No. : 2201020220
Enter Branch : AIML

Raj
2201020210
CSIT

Amit
2201020211
CSE

Itzu
2201020220
AIML
```

```c
// 2. WAP to Add Two Distances (in inch-feet system) using Structures.

#include <stdio.h>

struct Distance
{
    int feet;
    int inch;
};


int main(){
    struct Distance d[2];

    for (int i = 0; i < 2; i++)
    {
        printf("Enter feet : ");
        scanf("%d", &d[i].feet);

        printf("Enter inch : ");
        scanf("%d", &d[i].inch);

        printf("\n");
    }

    int feet1 = d[0].feet, inch1 = d[0].inch;

    int feet2 = d[1].feet, inch2 = d[1].inch;

    int feet = feet1+feet2, inch = inch1+inch2;

    if(inch>=12){
        feet += (inch/12);
        inch = inch%12;
    }

    printf("Total\n\nFeet : %d\nInch : %d", feet, inch);

    return 0;
}
```

Output :-

```
Enter feet : 5
Enter inch : 10

Enter feet : 5
Enter inch : 12

Total

Feet : 11
Inch : 10
```

```c
// 3. WAP to Demonstrate the Dynamic Memory Allocation for Structure.

#include <stdio.h>
#include <stdlib.h>

// Define the structure
struct person {
    char name[20];
    int age;
};

int main() {
    // Declare a pointer to a person struct
    struct person *p;

    // Dynamically allocate memory for the person struct
    p = (struct person*) malloc(sizeof(struct person));

    // Check if memory allocation was successful
    if (p == NULL) {
        printf("Memory allocation failed.");
        return 1;
    }

    // Populate the person struct
    printf("Enter the person's name: ");
    scanf("%s", p->name);
    printf("Enter the person's age: ");
    scanf("%d", &p->age);

    // Display the person's information
    printf("\nName: %s", p->name);
    printf("\nAge: %d", p->age);

    // Free the dynamically allocated memory
    free(p);

    return 0;
}
```

Output :

```
Enter the person's name: RAJ
Enter the person's age: 25

Name: RAJ
Age: 25
```