Structure Basics assignments

1. WAP to define read and store the following details together in a structure and display the structure details
name – char* [input could of maximum length 50 char]
age – uint
designation – enum [use the enum values used in enum exercise]
void read (EMP *emp)
void display (EMP emp);
Implement additional function below.
//read and update the employee record
int update(EMP *emp);
//copy the emp to a new employee and return
EMP copy(EMP emp);

Sol:

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

enum designation {
  Manager = 1,
  Developer,
  HR,
  Intern
};

struct EMP {
  char name[50];
  unsigned int age;
  enum designation desig;
};

void read(struct EMP *emp) {
```

```
    printf("Enter employee name: ");
    fgets(emp->name, sizeof(emp->name), stdin);
    emp->name[strcspn(emp->name, "\n")] = '\0';

    printf("Enter employee age: ");
    scanf("%u", &emp->age);

    printf("Enter employee designation (1 = Manager, 2 = Developer, 3 = HR, 4 = Intern): ");
    int desig;
    scanf("%d", &desig);
    emp->desig = (enum designation) desig;
}

void display(struct EMP emp) {
    char *designation_str;
    switch (emp.desig) {
        case Manager:
            designation_str = "Manager";
            break;
        case Developer:
            designation_str = "Developer";
            break;
        case HR:
            designation_str = "HR";
            break;
        case Intern:
            designation_str = "Intern";
            break;
        default:
            designation_str = "Unknown";
    }
```

```
    printf("Employee Details:\n");
    printf("Name: %s\n", emp.name);
    printf("Age: %u\n", emp.age);
    printf("Designation: %s\n", designation_str);
}

int update(struct EMP *emp) {
    printf("Update employee details\n");

    printf("Enter new name: ");
    getchar();
    fgets(emp->name, sizeof(emp->name), stdin);
    emp->name[strcspn(emp->name, "\n")] = '\0';

    printf("Enter new age: ");
    scanf("%u", &emp->age);

    printf("Enter new designation (1 = Manager, 2 = Developer, 3 = HR, 4 = Intern): ");
    int desig;
    scanf("%d", &desig);
    emp->desig = (enum designation) desig;

    return 1;
}
struct EMP copy(struct EMP emp) {
    struct EMP new_emp;
    strcpy(new_emp.name, emp.name);
    new_emp.age = emp.age;
    new_emp.desig = emp.desig;
    return new_emp;
```

```
}
int main() {
   struct EMP emp1;
    read(&emp1);
  display(emp1);
   update(&emp1);
   display(emp1);
   struct EMP emp2 = copy(emp1);
    printf("\nCopied Employee Details:\n");
   display(emp2);


   return 0;
}
```

Output:

```
Enter employee designation (1 = Manager, 2 = Developer, 3 = HR, 4 = Intern): 2
Employee Details:
Name: Iswarya
Age: 20
Designation: Developer
Update employee details
Enter new name: Pothala
Enter new age: 21
Enter new designation (1 = Manager, 2 = Developer, 3 = HR, 4 = Intern): 2
Employee Details:
Name: Pothala
Age: 21
Designation: Developer

Copied Employee Details:
Employee Details:
Name: Pothala
Age: 21
```

2. Extend the above program and create and initialize an array of 3 structures. Reuse the read() and display() functions to read, initialize and display structures.

Sol:

```
#include <stdio.h>

#include <stdlib.h>

#include <string.h>
```

```
enum designation {
   Manager = 1,
   Developer,
   HR,
   Intern
};

struct EMP {
   char name[50];
   unsigned int age;
   enum designation desig;
};

void read(struct EMP *emp) {
   printf("Enter employee name: ");
   fgets(emp->name, sizeof(emp->name), stdin);
   emp->name[strcspn(emp->name, "\n")] = '\0';
    getchar();

   printf("Enter employee age: ");
   scanf("%u", &emp->age);
    getchar();
   printf("Enter employee designation (1 = Manager, 2 = Developer, 3 = HR, 4 = Intern): ");
   int desig;
   scanf("%d", &desig);
   getchar();
   emp->desig = (enum designation) desig;
}
void display(struct EMP emp) {
   char *designation_str;
   switch (emp.desig) {
```

```
        case Manager:
            designation_str = "Manager";
            break;
        case Developer:
            designation_str = "Developer";
            break;
        case HR:
            designation_str = "HR";
            break;
        case Intern:
            designation_str = "Intern";
            break;
        default:
            designation_str = "Unknown";
    }

    printf("Employee Details:\n");
    printf("Name: %s\n", emp.name);
    printf("Age: %u\n", emp.age);
    printf("Designation: %s\n", designation_str);
}

int update(struct EMP *emp) {
    printf("Update employee details\n");

    printf("Enter new name: ");
    getchar();
    fgets(emp->name, sizeof(emp->name), stdin);
    emp->name[strcspn(emp->name, "\n")] = '\0';
    printf("Enter new age: ");
    scanf("%u", &emp->age);
```

```
    printf("Enter new designation (1 = Manager, 2 = Developer, 3 = HR, 4 = Intern): ");
    int desig;
    scanf("%d", &desig);
    emp->desig = (enum designation) desig;

    return 1;
}
struct EMP copy(struct EMP emp) {
    struct EMP new_emp;
    strcpy(new_emp.name, emp.name);
    new_emp.age = emp.age;
    new_emp.desig = emp.desig;
    return new_emp;
}

int main() {
    struct EMP employees[3];
    for (int i = 0; i < 3; i++) {
        printf("\nReading details for Employee %d\n", i + 1);
        read(&employees[i]); // Read employee details
    }
    for (int i = 0; i < 3; i++) {
        printf("\nDisplaying details for Employee %d\n", i + 1);
        display(employees[i]);
    }
    printf("\nUpdating details for Employee 1:\n");
    update(&employees[0]);

    printf("\nUpdated details for Employee 1:\n");
    display(employees[0]);
```

```
    struct EMP copied_employee = copy(employees[1]);

    printf("\nCopied Employee Details:\n");
    display(copied_employee);

    return 0;
}
```

Output: