***Assignment 14 || Structure***

*Arjun Patel – FRN-13J1124/006*

***Note: I mistakenly not uploaded part 2 and 3 of assignment 13 so, I am submitting it here.***

***Assignment 13 remaining part:***

Q1) Pass by Value using function

#include<stdio.h>

#include<string.h>

#define s(x) x\*x

typedef struct Employee{

    int id;

    char name[30];

    int salary;

} Employee;

void storeEmps(Employee\* emp, int n){

    for (int i = 0; i < n; i++)

    {

        printf("Enter id\n");

        scanf("%d", &emp[i].id);

        printf("Enter name\n");

        scanf("%s", emp[i].name);

        printf("Enter salary\n");

        scanf("%d", &emp[i].salary);

    }

}

void printEmps(Employee\* emp, int n){

    for (int i = 0; i < n; i++)

    {

        printf("\nid-> %d  Name-> %s  Salary->%d\n", emp[i].id, emp[i].name, emp[i].salary);

    }

}

int main(){

    Employee arr[3];

    storeEmps(arr, 3);

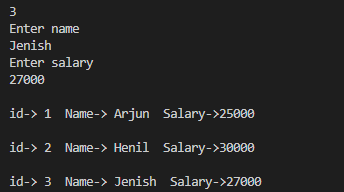
    printEmps(arr, 3);

    // printf("%d", s(5));

    // printf("%d", s(3+2));

    return 0;

}



Q3)

#include<stdio.h>

#include<string.h>

typedef struct Admin{

    int id;

    char name[30];

    int salary;

    int allowance;

} Admin;

void storeStrct(Admin\* emp, int n){

    for (int i = 0; i < n; i++)

    {

        printf("Enter id\n");

        scanf("%d", &emp[i].id);

        printf("Enter name\n");

        scanf("%s", emp[i].name);

        printf("Enter salary\n");

        scanf("%d", &emp[i].salary);

        printf("Enter allowance\n");

        scanf("%d", &emp[i].allowance);

    }

}

void printStrct(Admin\* emp, int n){

    for (int i = 0; i < n; i++)

    {

        printf("\nid-> %d  Name-> %s  Salary->%d Allowance->%d\n", emp[i].id, emp[i].name, emp[i].salary, emp[i].allowance);

    }

}

int main(){

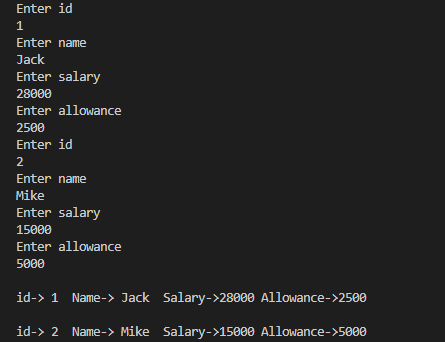
    struct Admin arr[2];

    storeStrct(arr, 2);

    printStrct(arr, 2);

    return 0;

}



Q4)

#include<stdio.h>

#include<string.h>

typedef struct HR{

    int id;

    char name[30];

    int salary;

    int commision;

} HR;

void storeStrct(HR\* h, int n){

    for (int i = 0; i < n; i++)

    {

        printf("Enter id\n");

        scanf("%d", &h[i].id);

        printf("Enter name\n");

        scanf("%s", h[i].name);

        printf("Enter salary\n");

        scanf("%d", &h[i].salary);

        printf("Enter Commision\n");

        scanf("%d", &h[i].commision);

    }

}

void printStrct(HR\* h, int n){

    for (int i = 0; i < n; i++)

    {

        printf("\nid-> %d  Name-> %s  Salary->%d Commision->%d\n", h[i].id, h[i].name, h[i].salary, h[i].commision);

    }

}

int main(){

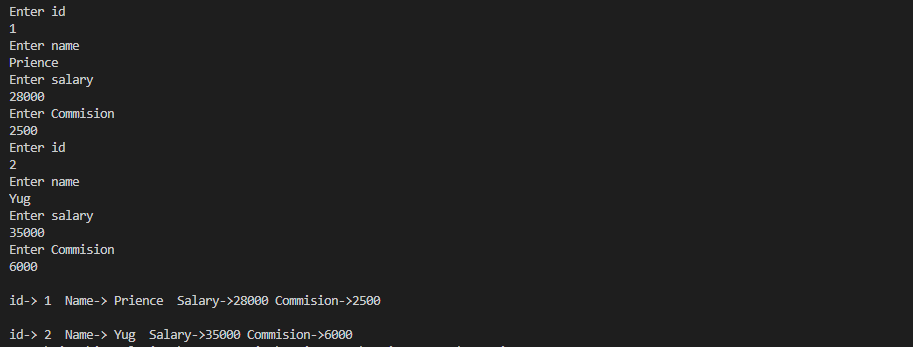
    struct HR h[2];

    storeStrct(h, 2);

    printStrct(h, 2);

    return 0;

}



Q5)

#include<stdio.h>

#include<string.h>

typedef struct SalesManager{

    int id;

    char name[30];

    int salary;

    int incentive;

    int target;

} SalesManager;

void storeStrct(SalesManager\* h, int n){

    for (int i = 0; i < n; i++)

    {

        printf("Enter id\n");

        scanf("%d", &h[i].id);

        printf("Enter name\n");

        scanf("%s", h[i].name);

        printf("Enter salary\n");

        scanf("%d", &h[i].salary);

        printf("Enter Incentive\n");

        scanf("%d", &h[i].incentive);

        printf("Enter Target\n");

        scanf("%d", &h[i].target);

    }

}

void printStrct(SalesManager\* h, int n){

    for (int i = 0; i < n; i++)

    {

        printf("\nid-> %d  Name-> %s  Salary->%d Incentive->%d Target->%d\n", h[i].id, h[i].name, h[i].salary, h[i].incentive, h[i].target);

    }

}

int main(){

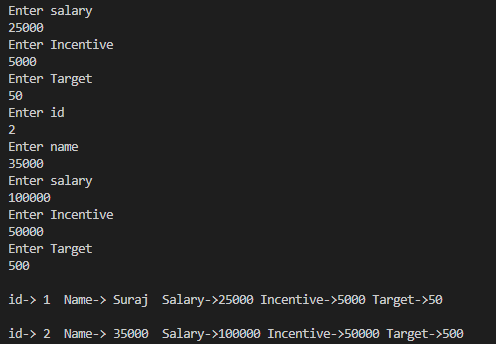
    struct SalesManager smr[2];

    storeStrct(smr, 2);

    printStrct(smr, 2);

    return 0;

}



Q6)

#include<stdio.h>

#include<string.h>

typedef struct Date{

    int date;

    int month;

    int year;

} Date;

void storeStrct(Date\* dt, int n){

    for (int i = 0; i < n; i++)

    {

        printf("Enter date\n");

        scanf("%d", &dt[i].date);

        printf("Enter month\n");

        scanf("%d", &dt[i].month);

        printf("Enter year\n");

        scanf("%d", &dt[i].year);

    }

}

void printStrct(Date\* dt, int n){

    for (int i = 0; i < n; i++)

    {

        printf("\n%d/%d/%d\n", dt[i].date, dt[i].month, dt[i].year);

    }

}

int main(){

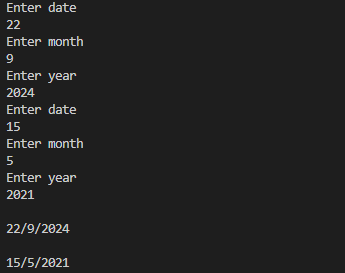
    struct Date dt[2];

    storeStrct(dt, 2);

    printStrct(dt, 2);

    return 0;

}



Q7)

#include<stdio.h>

#include<string.h>

typedef struct Time{

    int hr;

    int min;

    int sec;

} Time;

void storeStrct(Time\* dt, int n){

    for (int i = 0; i < n; i++)

    {

        printf("Enter Hour\n");

        scanf("%d", &dt[i].hr);

        printf("Enter Minutes\n");

        scanf("%d", &dt[i].min);

        printf("Enter Seconds\n");

        scanf("%d", &dt[i].sec);

    }

}

void printStrct(Time\* dt, int n){

    for (int i = 0; i < n; i++)

    {

        printf("\nHr-> %d Minutes->%d Sec->%d\n", dt[i].hr, dt[i].min, dt[i].sec);

    }

}

int main(){

    struct Time t[2];

    t[1].hr = 2;

    t[1].min = 20;

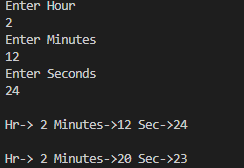
    t[1].sec = 23;

    storeStrct(t, 1);

    printStrct(t, 2);

    return 0;

}



Q8)

#include<stdio.h>

#include<string.h>

typedef struct Distance{

    float feet;

    float inch;

} Distance;

void storeStrct(Distance\* dt, int n){

    for (int i = 0; i < n; i++)

    {

        printf("Enter Feet\n");

        scanf("%f", &dt[i].feet);

        printf("Enter Inch\n");

        scanf("%f", &dt[i].inch);

    }

}

void printStrct(Distance\* dt, int n){

    for (int i = 0; i < n; i++)

    {

        printf("\nLength is -> %.2f feets %.2f  inchs\n", dt[i].feet, dt[i].inch);

        printf("OR\n");

        printf("Length is -> %.2f' %.2f''", dt[i].feet, dt[i].inch);

    }

}

int main(){

    Distance t[2];

    t[1].feet = 2;

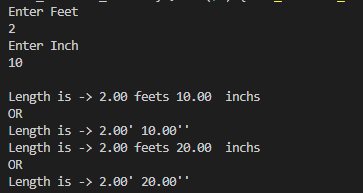
    t[1].inch = 20;

    storeStrct(t, 1);

    printStrct(t, 2);

    return 0;

}



Q9)

#include<stdio.h>

#include<string.h>

typedef struct Complex{

    int real;

    int img;

} Complex;

void storeStrct(Complex\* cmp, int n){

    for (int i = 0; i < n; i++)

    {

        printf("Enter real num\n");

        scanf("%d", &cmp[i].real);

        printf("Enter imaginary num\n");

        scanf("%d", &cmp[i].img);

    }

}

void printStrct(Complex\* cmp, int n){

    for (int i = 0; i < n; i++)

    {

        printf("\n%d + %di\n", cmp[i].real, cmp[i].img);

    }

}

int main(){

    struct Complex cmp[2];

    cmp[1].real = 2;

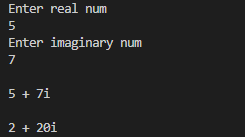
    cmp[1].img = 20;

    storeStrct(cmp, 1);

    printStrct(cmp, 2);

    return 0;

}



Q10)

#include<stdio.h>

#include<string.h>

typedef struct Product{

    int id;

    char name[30];

    int quantity;

    int price;

} Product;

void storeStrct(Product\* prdt, int n){

    for (int i = 0; i < n; i++)

    {

        printf("Enter product id\n");

        scanf("%d", &prdt[i].id);

        printf("Enter product name\n");

        scanf("%s", prdt[i].name);

        printf("Enter quantity\n");

        scanf("%d", &prdt[i].quantity);

        printf("Enter price\n");

        scanf("%d", &prdt[i].price);

    }

}

void printStrct(Product\* prdt, int n){

    for (int i = 0; i < n; i++)

    {

        printf("\nid-> %d  Name-> %s  quantity->%d  price->%d totalAmount->%d rs.\n", prdt[i].id, prdt[i].name, prdt[i].quantity, prdt[i].price, prdt[i].quantity \* prdt[i].price);

    }

}

int main(){

    Product p[2];

    p[1].id = 2;

    strcpy(p[1].name, "Milk");

    p[1].quantity = 5;

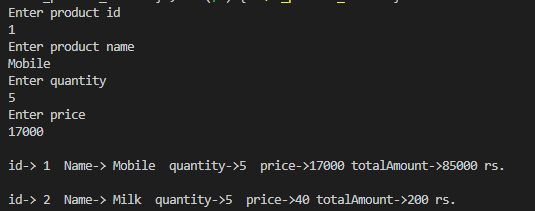
    p[1].price = 40;

    storeStrct(p, 1);

    printStrct(p, 2);

    return 0;

}



***------END------***

**Part 3:** Pass by reference

Q1)

#include<stdio.h>

#include<string.h>

typedef struct Student{

    int rollNo;

    char name[30];

    int marks[3]; ///marks of two subjects

} Student;

void storeStudent(Student\* s1){

    printf("Enter roll no\n");

    scanf("%d", &s1->rollNo);

    printf("Enter name no\n");

    scanf("%s", &s1->name);

    for (int i = 0; i < 3; i++)

    {

        printf("Enter marks for %d\n", i+1);

        scanf("%d", &s1->marks[i]);

    }

}

int main(){

    Student s1, s2;

    storeStudent(&s1);

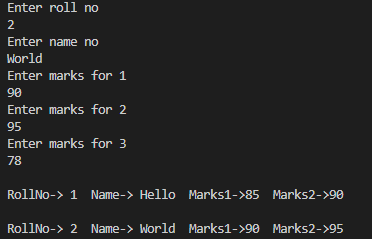
    storeStudent(&s2);

    printf("\nRollNo-> %d  Name-> %s  Marks1->%d  Marks2->%d\n", s1.rollNo, s1.name, s1.marks[0], s1.marks[1]);

    printf("\nRollNo-> %d  Name-> %s  Marks1->%d  Marks2->%d\n", s2.rollNo, s2.name, s2.marks[0], s2.marks[1]);

    return 0;

}



Q2)

#include<stdio.h>

#include<string.h>

typedef struct Employee{

    int id;

    char name[30];

    int salary;

} Employee;

void storeEmps(Employee\* emp){

        printf("Enter id\n");

        scanf("%d", &emp->id);

        printf("Enter name\n");

        scanf("%s", emp->name);

        printf("Enter salary\n");

        scanf("%d", &emp->salary);

}

void printEmps(Employee\* emp){

        printf("\nid-> %d  Name-> %s  Salary->%d\n", emp->id, emp->name, emp->salary);

}

int main(){

    Employee e1, e2, e3;

    storeEmps(&e1);

    storeEmps(&e2);

    storeEmps(&e3);

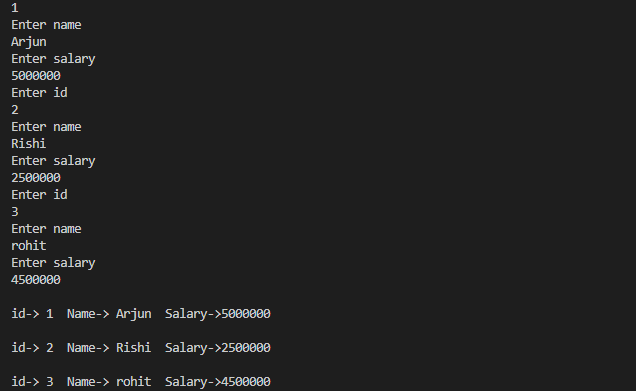
    printEmps(&e1);

    printEmps(&e2);

    printEmps(&e3);

    return 0;

}



Q3)

#include<stdio.h>

#include<string.h>

typedef struct HR{

    int id;

    char name[30];

    int salary;

    int commision;

} HR;

void storeStrct(HR\* h){

        printf("Enter id\n");

        scanf("%d", &h->id);

        printf("Enter name\n");

        scanf("%s", h->name);

        printf("Enter salary\n");

        scanf("%d", &h->salary);

        printf("Enter Commision\n");

        scanf("%d", &h->commision);

}

void printStrct(HR\* h){

        printf("\nid-> %d  Name-> %s  Salary->%d Commision->%d\n", h->id, h->name, h->salary, h->commision);

}

int main(){

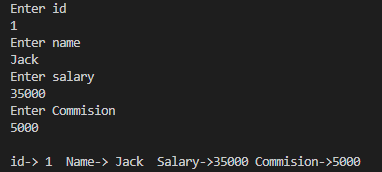
    struct HR h;

    storeStrct(&h);

    printStrct(&h);

    return 0;

}



Q4)

#include<stdio.h>

#include<string.h>

typedef struct Admin{

    int id;

    char name[30];

    int salary;

    int allowance;

} Admin;

void storeStrct(Admin\* admn){

        printf("Enter id\n");

        scanf("%d", &admn->id);

        printf("Enter name\n");

        scanf("%s", admn->name);

        printf("Enter salary\n");

        scanf("%d", &admn->salary);

        printf("Enter allowance\n");

        scanf("%d", &admn->allowance);

}

void printStrct(Admin\* admn){

        printf("\nid-> %d  Name-> %s  Salary->%d Allowance->%d\n", admn->id, admn->name, admn->salary, admn->allowance);

}

int main(){

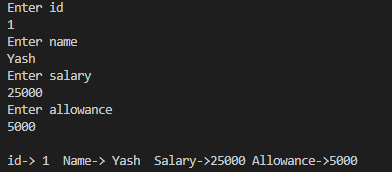
    struct Admin a1;

    storeStrct(&a1);

    printStrct(&a1);

    return 0;

}



Q5)

#include<stdio.h>

#include<string.h>

typedef struct SalesManager{

    int id;

    char name[30];

    int salary;

    int incentive;

    int target;

} SalesManager;

void storeStrct(SalesManager\* sm){

        printf("Enter id\n");

        scanf("%d", &sm->id);

        printf("Enter name\n");

        scanf("%s", sm->name);

        printf("Enter salary\n");

        scanf("%d", &sm->salary);

        printf("Enter Incentive\n");

        scanf("%d", &sm->incentive);

        printf("Enter Target\n");

        scanf("%d", &sm->target);

}

void printStrct(SalesManager\* sm){

        printf("\nid-> %d  Name-> %s  Salary->%d Incentive->%d Target->%d\n", sm->id, sm->name, sm->salary, sm->incentive, sm->target);

}

int main(){

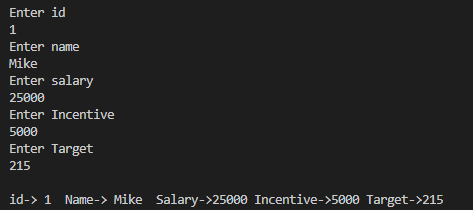
    struct SalesManager smr;

    storeStrct(&smr);

    printStrct(&smr);

    return 0;

}



Q6)

#include <stdio.h>

#include <string.h>

typedef struct Date

{

    int date;

    int month;

    int year;

} Date;

void storeStrct(Date \*dt)

{

    printf("Enter date\n");

    scanf("%d", &dt->date);

    printf("Enter month\n");

    scanf("%d", &dt->month);

    printf("Enter year\n");

    scanf("%d", &dt->year);

}

void printStrct(Date \*dt)

{

    printf("\n%d/%d/%d\n", dt->date, dt->month, dt->year);

}

int main()

{

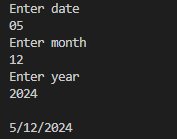
    struct Date dt;

    storeStrct(&dt);

    printStrct(&dt);

    return 0;

}



Q7)

#include<stdio.h>

#include<string.h>

typedef struct Time{

    int hr;

    int min;

    int sec;

} Time;

void storeStrct(Time\* dt){

        printf("Enter Hour\n");

        scanf("%d", &dt->hr);

        printf("Enter Minutes\n");

        scanf("%d", &dt->min);

        printf("Enter Seconds\n");

        scanf("%d", &dt->sec);

}

void printStrct(Time\* dt){

        printf("\nHr-> %d Minutes->%d Sec->%d\n", dt->hr, dt->min, dt->sec);

}

int main(){

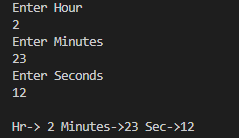
    struct Time t;

    storeStrct(&t);

    printStrct(&t);

    return 0;

}



Q8)

#include <stdio.h>

#include <string.h>

typedef struct Distance

{

    float feet;

    float inch;

} Distance;

void storeStrct(Distance \*dt)

{

    printf("Enter Feet\n");

    scanf("%f", &dt->feet);

    printf("Enter Inch\n");

    scanf("%f", &dt->inch);

}

void printStrct(Distance \*dt)

{

    printf("\nLength is -> %.2f feets %.2f  inchs\n", dt->feet, dt->inch);

    printf("OR\n");

    printf("Length is -> %.2f' %.2f''", dt->feet, dt->inch);

}

int main()

{

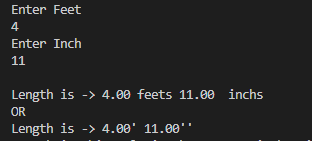
    Distance t;

    storeStrct(&t);

    printStrct(&t);

    return 0;

}



Q9)

#include<stdio.h>

#include<string.h>

typedef struct Complex{

    int real;

    int img;

} Complex;

void storeStrct(Complex\* cmp){

        printf("Enter real num\n");

        scanf("%d", &cmp->real);

        printf("Enter imaginary num\n");

        scanf("%d", &cmp->img);

}

void printStrct(Complex\* cmp){

        printf("\n%d + %di\n", cmp->real, cmp->img);

}

int main(){

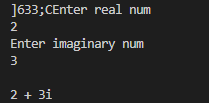
    struct Complex cmp;

    storeStrct(&cmp);

    printStrct(&cmp);

    return 0;

}



Q10)

#include<stdio.h>

#include<string.h>

typedef struct Product{

    int id;

    char name[30];

    int quantity;

    int price;

} Product;

void storeStrct(Product\* prdt){

        printf("Enter product id\n");

        scanf("%d", &prdt->id);

        printf("Enter product name\n");

        scanf("%s", prdt->name);

        printf("Enter quantity\n");

        scanf("%d", &prdt->quantity);

        printf("Enter price\n");

        scanf("%d", &prdt->price);

}

void printStrct(Product\* prdt){

        printf("\nid-> %d  Name-> %s  quantity->%d  price->%d totalAmount->%d rs.\n", prdt->id, prdt->name, prdt->quantity, prdt->price, prdt->quantity \* prdt->price);

}

int main(){

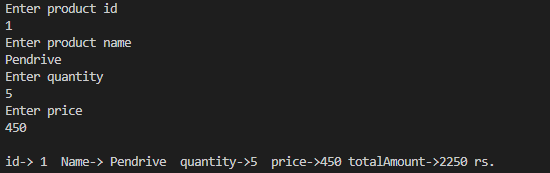
    Product p;

    storeStrct(&p);

    printStrct(&p);

    return 0;

}



**Assignment 14 continue:**

Q1) Create a structure Book with data members as name, id, author, price. Accept the values of all these members from user and display them.

#include<stdio.h>

#include<string.h>

typedef struct Book{

    char name[30];

    int id;

    char author[30];

    double price;

} Book;

void fgetsInput(char\* str, int size){

    int len = strlen(str);

    fflush(stdin);

    if(fgets(str, size, stdin)){

        if(len>0 && str[len-1]=='\n') str[len-1] = '\0';

    }

    fflush(stdin);

}

void displayAll(Book\* books,int n){

    for (int i = 0; i < n; i++)

    {

        printf("Book Name -> %s", books[i].name);

        printf("Book id -> %d\n", books[i].id);

        printf("Author Name -> %s", books[i].author);

        printf("Book id -> %.2lf\n", books[i].price);

    }

}

int main(){

    Book books[3];

    for (int i = 0; i < 3; i++)

    {

        printf("---Book %d---\n", i+1);

        printf("Enter name of the book\n");

        fgetsInput(books[i].name, sizeof(books[i].name));

        printf("Enter id of book\n");

        scanf("%d", &books[i].id);

        printf("Enter name of the author\n");

        fgetsInput(books[i].author, sizeof(books[i].name));

        printf("Enter price of book\n");

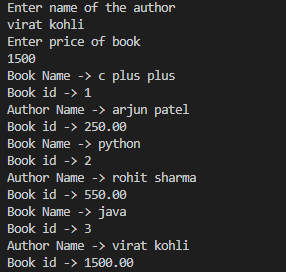
        scanf("%lf", &books[i].price);

    }

    displayAll(books, 3);

    return 0;

}



Q2)

#include<stdio.h>

typedef struct Time{

    int hr;

    int min;

    char sec;

} Time;

void displayAll(Time\* t,int n){

    for (int i = 0; i < n; i++)

    {

        printf("%d : %d : %d\n", t[i].hr, t[i].min, t[i].sec);

    }

}

int convertIntoSecs(Time t){

    int total = (t.hr\*3600) + (t.min\*60) + (t.sec);

    return total;

}

void add(Time t1, Time t2){

    int totalSecs = convertIntoSecs(t1) + convertIntoSecs(t2);

    int totalHrs =  totalSecs/3600;

    int hrRem = totalSecs%3600;

    int totalMin =  hrRem/60;

    int minRem = hrRem%60;

    int totalSec =  minRem%60;

    printf("%d : %d : %d\n", totalHrs, totalMin, totalSec);

}

int main(){

    Time t[2];

    for (int i = 0; i < 2; i++)

    {

        printf("---Time %d---\n", i+1);

        printf("Enter hrs\n");

        scanf("%d", &t[i].hr);

        printf("Enter minutes\n");

        scanf("%d", &t[i].min);

        printf("Enter secs\n");

        scanf("%d", &t[i].sec);

    }

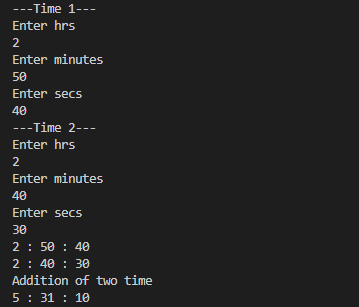
    displayAll(t, 2);

    printf("Addition of two time\n");

    add(t[0], t[1]);

    return 0;

}



Q3) Write a program to create an array for 10 players. For each player store name, no. of matches played, runs, wickets takes.

a. Create function to Accept the information of each player.

b. Create function to display the information of all the players

c. Display the information of player who made maximum runs and the one who took maximum number of wickets.

#include <stdio.h>

#include <string.h>

/\*

Write a program to create an array for 10 players. For each player store name, no. of matches played, runs, wickets takes.

a. Create function to Accept the information of each player.

b. Create function to display the information of all the players

c. Display the information of player who made maximum runs and the one who took maximum number of wickets.

\*/

typedef struct Players

{

    char name[30];

    int noOfMatches;

    int runs;

    int wickets;

} Players;

void fgetsInput(char \*str, size\_t size)

{

    fflush(stdin);

    if (fgets(str, size, stdin))

    {

        // Remove newline character from fgets

        int len = strlen(str);

        if (len > 0 && str[len - 1] == '\n')

        {

            str[len - 1] = '\0';

        }

    }

}

void displayAll(Players \*p, int n)

{

    for (int i = 0; i < n; i++)

    {

        printf("Name -> %s | ", p[i].name);

        printf("Matchs Played -> %d | ", p[i].noOfMatches);

        printf("Runs -> %d | ", p[i].runs);

        printf("Wickets -> %d\n", p[i].wickets);

    }

}

void displayByIndex(Players \*p, int i)

{

    printf("Name -> %s | ", p[i].name);

    printf("Matchs Played -> %d | ", p[i].noOfMatches);

    printf("Runs -> %d | ", p[i].runs);

    printf("Wickets -> %d\n", p[i].wickets);

}

void storeStruct(Players \*p, int n)

{

    for (int i = 0; i < n; i++)

    {

        printf("---Players %d---\n", i + 1);

        printf("Enter name of the Players\n");

        fgetsInput(p[i].name, sizeof(p[i].name));

        printf("Enter No of matching played\n");

        scanf("%d", &p[i].noOfMatches);

        printf("Enter runs score by player\n");

        scanf("%d", &p[i].runs);

        printf("Enter wickets taken by player\n");

        scanf("%d", &p[i].wickets);

    }

}

void displayTableToppers(Players \*p, int n)

{

    int maxRuns = p[0].runs, maxWicket = p[0].wickets;

    int maxRunsIndex = 0, maxWicketIndex = 0;

    for (int i = 0; i < n; i++)

    {

        if (p[i].runs > maxRuns)

        {

            maxRuns = p[i].runs;

            maxRunsIndex = i;

        }

        if (p[i].wickets > maxWicket)

        {

            maxRuns = p[i].wickets;

            maxWicketIndex = i;

        }

    }

    printf("Player who score maximum runs\n");

    displayByIndex(p, maxRunsIndex);

    printf("Player who taken maximum wickets\n");

    displayByIndex(p, maxWicketIndex);

}

int main()

{

    Players p[3];

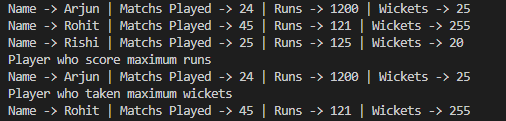
    storeStruct(p, 3);

    displayAll(p, 3);

    displayTableToppers(p, 3);

    return 0;

}



Q4) Point of Sale System: Build a simple point of sale system using structures to represent products with attributes like name, price, and quantity. Allow users to add items to a cart and calculate the total cost.

#include <stdio.h>

#include <string.h>

/\*

Point of Sale System: Build a simple point of sale system using structures to represent products with attributes like name, price, and quantity. Allow users to add items to a cart and calculate the total cost.

\*/

typedef struct Products

{

    int id;

    char name[30];

    double price;

} Products;

typedef struct CartItems

{

    Products p1;

    int quantity;

} CartItems;

void fgetsInput(char \*str, size\_t size)

{

    fflush(stdin);

    if (fgets(str, size, stdin))

    {

        // Remove newline character from fgets

        int len = strlen(str);

        if (len > 0 && str[len - 1] == '\n')

        {

            str[len - 1] = '\0';

        }

    }

}

void displayAll(Products \*p, int n)

{

    printf("----------------------------------\n");

    printf("|  Id  |    Product     | Price  |\n");

    printf("|------|----------------|--------|\n");

    for (int i = 0; i < n; i++)

    {

        int len = strlen(p[i].name);

        printf("|  %d   ", p[i].id);

        printf("| %s", p[i].name);

        for (int i = 1; i <= 15 - len; i++)

        {

            printf(" ");

        }

        i == 3 && printf("| %.2lf |\n", p[i].price);

        i != 3 && printf("| %.2lf  |\n", p[i].price);

    }

    printf("----------------------------------\n");

}

// void displayByIndex(Products \*p, int i)

// {

//     printf("Name -> %s | ", p[i].name);

//     printf("Matchs Played -> %d | ", p[i].noOfMatches);

//     printf("Runs -> %d | ", p[i].runs);

//     printf("Wickets -> %d\n", p[i].wickets);

// }

// void storeStruct(Players \*p, int n)

// {

//     for (int i = 0; i < n; i++)

//     {

//         printf("---Players %d---\n", i + 1);

//         printf("Enter name of the Players\n");

//         fgetsInput(p[i].name, sizeof(p[i].name));

//         printf("Enter No of matching played\n");

//         scanf("%d", &p[i].noOfMatches);

//         printf("Enter runs score by player\n");

//         scanf("%d", &p[i].runs);

//         printf("Enter wickets taken by player\n");

//         scanf("%d", &p[i].wickets);

//     }

// }

void generateAndDisplayBill(CartItems \*c, int\* n)

{

    double totalAmount = 0;

    printf("-------------------------------------------------------\n");

    printf("|  Id  |    Product     | Price  | Quantity |  Total  |\n");

    printf("|------|----------------|--------|----------|---------|\n");

    for (int i = 0; i < \*n; i++)

    {

        int len = strlen(c[i].p1.name);

        printf("|  %d   ", i+1);

        printf("| %s", c[i].p1.name);

        for (int i = 1; i <= 15 - len; i++)

        {

            printf(" ");

        }

        // printf("| %.2lf  \n", c[i].p1.price);

        c[i].p1.id == 3 && printf("| %.2lf ", c[i].p1.price);

        c[i].p1.id != 3 && printf("| %.2lf  ", c[i].p1.price);

        printf("|  %d      ", c[i].quantity);

        printf("| %.2lf |\n", c[i].quantity \* c[i].p1.price);

        totalAmount += c[i].quantity \* c[i].p1.price;

    }

    printf("-------------------------------------------------------\n");

    printf("                     Total Amount to pay ---> %.2lf\n\n\n", totalAmount);

}

void addToCart(CartItems \*c, Products \*p, int \*i)

{

    int choice=1, pid;

    while (choice)

    {

        printf("Enter Product id of product want to add to cart\n");

        scanf("%d", &pid);

        //adding product at index sent from main

        c[\*i].p1 = p[pid - 1];

        printf("Enter quantity\n");

        printf("%d %d  \n", \*i , c[\*i].quantity);

        (\*i)++;

        printf("\nPress 1 to Add more items\n");

        printf("Press 0 to generate bill\n");

        scanf("%d", &choice);

        if(choice) displayAll(p,5);

    }

    generateAndDisplayBill(c, &(\*i));

}

void hardCodedProducts(Products \*p)

{

    p[0].id = 1;

    strcpy(p[0].name, "Milk");

    p[0].price = 30;

    p[1].id = 2;

    strcpy(p[1].name, "Pen");

    p[1].price = 10;

    p[2].id = 3;

    strcpy(p[2].name, "Chips");

    p[2].price = 20;

    p[3].id = 4;

    strcpy(p[3].name, "Washing Powder");

    p[3].price = 150;

    p[4].id = 5;

    strcpy(p[4].name, "Soap");

    p[4].price = 20;

}

int main()

{

    int cartItemIndex = 0;

    Products p[5];

    CartItems c[50];

    // storeStruct(p, 3);

    hardCodedProducts(p);

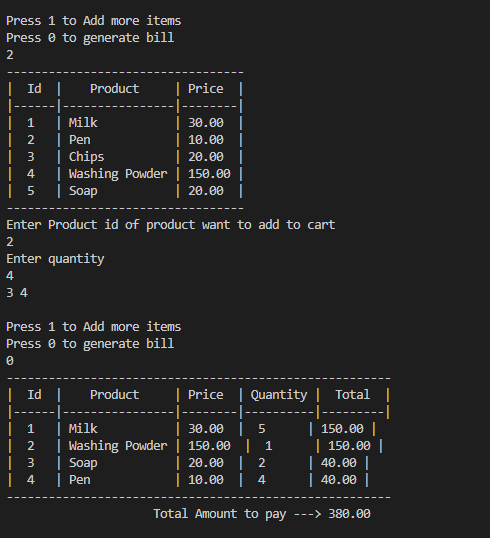
    displayAll(p, 5);

    addToCart(c, p, &cartItemIndex);

    // displayTableToppers(p, 3);

    return 0;

}



Q5) Movie Database: Create a program that uses structures to manage a movie database with details like title, director, release year, and genre. Allow users to add, search for, and update movie records.

#include <stdio.h>

#include <string.h>

/\*

Movie Database: Create a program that uses structures to manage a movie database with details like title, director, release year, and genre. Allow users to add, search for, and update movie records.

\*/

typedef struct Movie

{

    int id;

    char title[30];

    char director[30];

    int releaseYear;

    char genre[30];

} Movie;

void hardCoded(Movie \*mrr)

{

    mrr[0].id = 1;

    strcpy(mrr[0].title, "Animal");

    strcpy(mrr[0].director, "Arjun Reddy Vanga");

    mrr[0].releaseYear = 2024;

    strcpy(mrr[0].genre, "Action Crime");

    mrr[1].id = 2;

    strcpy(mrr[1].title, "3 Idiots");

    strcpy(mrr[1].director, "Vidhu Vinod Chopra");

    mrr[1].releaseYear = 2014;

    strcpy(mrr[1].genre, "Drama");

    mrr[2].id = 3;

    strcpy(mrr[2].title, "Stree");

    strcpy(mrr[2].director, "Maddock Production");

    mrr[2].releaseYear = 2023;

    strcpy(mrr[2].genre, "Horror Comedy");

}

void displayMovies(Movie \*mrr, int \*index)

{

    printf("-------------------------------------------------------------------------------------------\n");

    printf("| %-3s | %-20s | %-20s | %-12s | %-20s |\n", "Id", "Title", "Director", "Release Year", "Genre");

    printf("-------------------------------------------------------------------------------------------\n");

    for (int i = 0; i < \*index; i++)

    {

        printf("| %-3d | %-20s | %-20s | %-12d | %-20s |\n", mrr[i].id, mrr[i].title, mrr[i].director, mrr[i].releaseYear, mrr[i].genre);

        printf("-------------------------------------------------------------------------------------------\n");

    }

}

int addMovie(Movie \*mrr, int \*index)

{

    static int staticId = 4;

    mrr[\*index].id = staticId;

    printf("Assigned id --> %d\n", staticId);

    printf("Enter Movie Title\n");

    fflush(stdin);

    fgets(mrr[\*index].title, sizeof(mrr[\*index].title), stdin);

    // Optional: Remove newline character

    mrr[\*index].title[strcspn(mrr[\*index].title, "\n")] = '\0';

    printf("Enter the name of director\n");

    fgets(mrr[\*index].director, sizeof(mrr[\*index].director), stdin);

    // Optional: Remove newline character

    mrr[\*index].director[strcspn(mrr[\*index].director, "\n")] = '\0';

    fflush(stdin);

    printf("Enter release year of movie\n");

    scanf("%d", &mrr[\*index].releaseYear);

    fflush(stdin);

    printf("Enter the genre of movie\n");

    fgets(mrr[\*index].genre, sizeof(mrr[\*index].genre), stdin);

    // Optional: Remove newline character

    mrr[\*index].genre[strcspn(mrr[\*index].genre, "\n")] = '\0';

    staticId++;

    return mrr[\*index].id;

}

int searchById(Movie \*mrr, int \*index, int tempId)

{

    for (int i = 0; i < \*index; i++)

    {

        if (mrr[i].id = tempId)

            return i;

    }

}

void searchByName(Movie \*mrr, int \*index)

{

    char str[30];

    Movie tempArr[10];

    int j = 0;

    fflush(stdin);

    printf("Enter name of Movie you want to search/update\n");

    fgets(str, sizeof(str), stdin);

    fflush(stdin);

    // Optional: Remove newline character

    str[strcspn(str, "\n")] = '\0';

    for (int i = 0; i < \*index; i++)

    {

        if (strstr(strlwr(mrr[i].title), strlwr(str)))

        {

            tempArr[j] = mrr[i];

            j++;

        }

    }

    if (j)

        displayMovies(tempArr, &j);

    else

        printf("No result found!\n");

}

void updateMovie(Movie \*mrr, int \*index)

{

    int tempId;

    searchByName(mrr, index);

    printf("Enter id of movie u want to update\n");

    scanf("%d", &tempId);

    int res = searchById(mrr, index, tempId);

    char tempName[30];

    int choice;

    printf("1. Update movie title\n");

    printf("2. Update movie director\n");

    printf("3. Update movie release year\n");

    printf("4. Update movie genre\n");

    scanf("%d", &choice);

    fflush(stdin);

    switch (choice)

    {

    case 1:

        printf("Enter Movie Title\n");

        fflush(stdin);

        fgets(tempName, sizeof(tempName), stdin);

        // Optional: Remove newline character

        tempName[strcspn(tempName, "\n")] = '\0';

        strcpy(mrr[res].title, tempName);

        break;

    case 2:

        printf("Enter new director name\n");

        fflush(stdin);

        fgets(tempName, sizeof(tempName), stdin);

        // Optional: Remove newline character

        tempName[strcspn(tempName, "\n")] = '\0';

        strcpy(mrr[res].director, tempName);

        break;

    case 3:

        int tempYr;

        printf("Enter new release year\n");

        scanf("%d", &tempYr);

        mrr[res].releaseYear = tempYr;

        break;

    case 4:

        printf("Enter new genre\n");

        fflush(stdin);

        fgets(tempName, sizeof(tempName), stdin);

        // Optional: Remove newline character

        tempName[strcspn(tempName, "\n")] = '\0';

        strcpy(mrr[res].genre, tempName);

        break;

    default:

        printf("Invalid Choice\n");

        break;

    }

}

int main()

{

    int mIndex = 3, choice;

    Movie mrr[5];

    hardCoded(mrr);

    while (1)

    {

        printf("1. Add Movie\n");

        printf("2. Display All Movies\n");

        printf("3. Search Movie by Name\n");

        printf("4. Update Movie\n");

        scanf("%d", &choice);

        switch (choice)

        {

        case 1:

        {

            int res = addMovie(mrr, &mIndex);

            if (res)

            {

                printf("Movie added successfully with id -> %d\n", res);

                mIndex++;

            }

            else

                printf("Error in adding book\n");

            break;

        }

        case 2:

        {

            displayMovies(mrr, &mIndex);

            break;

        }

        case 3:

        {

            searchByName(mrr, &mIndex);

            break;

        }

        case 4:

        {

            updateMovie(mrr, &mIndex);

            break;

        }

        default:

            break;

        }

    }

    return 0;

}

