1 WAP to display multiplication table from 1 to 10

```
#include <stdio.h>

int main()
{
   int i = 1, j;
   while(i <= 10){
      j = 1;
      while(j <= 10){

         printf("%d * %d = %d\t",i, j, i * j);
         j++;
      }
      printf("\n");
      i++;
   }

   return 0;
}
```

2)WAP todisplay the following patter mentioned below

```
*
* *
* * *
* * * *
* * * * *
```

```
#include <stdio.h>
int main() {
   int rows = 5;
   int i = 1;

   while (i <= rows) {
      int j = 1;
      while (j <= i) {
         printf("* ");
         j++;
      }
      printf("\n");
      i++;
   }

   return 0;
}
```

3) WAP todisplay the following patter mentioned below (full pyramid)

```
#include <stdio.h>
int main() {
    int rows = 5, space, k;
    int i = 1;

    while (i <= rows) {
        int spaces = rows - i;
        int j = 1;

        while (spaces > 0) {
            printf(" ");
            spaces--;
        }

        while (j <= i) {
            printf("* ");
            j++;
        }

        printf("\n");
        i++;
    }

    return 0;
}
```

4) Write a c program to print fibonacci series upto n terms

```
#include <stdio.h>

int main() {
    int n, t1 = 0, t2 = 1, next;

    printf("Enter the number of terms: ");
    scanf("%d", &n);

    printf("Fibonacci Series: ");

    for (int i = 1; i <= n; i++) {
        printf("%d ", t1);
        next = t1 + t2;
        t1 = t2;
        t2 = next;
    }
    return 0;
}
```

5) WAP to calculate sum of first n natural numbers

```
#include<stdio.h>
int main(){
    int n, sum = 0;
    printf("enter natural number ");
    scanf("%d",&n);

    for(int i =1; i <= n; i++){
        sum += i;
         }
         printf("\n");
         printf("sum of natural number till n is =%d",sum);
         return 0;
}
```

6) WAP to reverse a number using for loop

```
#include <stdio.h>

int main() {
    int num, reversed = 0;

    printf("Enter an integer: ");
    scanf("%d", &num);

    for (; num != 0; num /= 10) {
        int remainder = num % 10;
        reversed = reversed * 10 + remainder;
    }

    printf("Reversed number: %d\n", reversed);
    return 0;
}
```

7) assignment: WAP for guessing game

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

```
#include<time.h>

int main(){

   printf("this is a guessing game");
   printf("i have chosen a number between 0 and 20 which you might guess.\n");

   int random=0, guess = 0, noofguess;
   time_t t;
   srand((unsigned)time(&t));
   random=rand() % 21;

   for(noofguess = 5; noofguess >0; --noofguess){
      printf("you have %d tries left.\n",noofguess);
      printf("enter a  guess. ");
      scanf("%d",&guess);

      if(guess == random)
      {
         printf("congratulations. you guessed it ");
      }
      else
      if(guess > random )
      printf("sorry, %d is wrong.my number is less than that");
      else if(guess < random )
      printf("sorry, %d is wrong.my number is greater than that");
   }


   return 0;
   }
```

8) WAP that prompts the user to enter a series of integers(upto a maximum of 20). The program should calculate and display the sum of all even numbers entered while skipping any negative numbers. Use the continue statement to skip processing for negative numbers.

```
#include<stdio.h>
int main(){
   int num, sum=0,count = 0;
   while(count < 20){
         printf("enter upto 20 integers(enter -1 to stop)\n");
         scanf("%d",&num);

         if(num == -1){
            break;
            }
            if(num < 0){
                continue;
            }
            if(num % 2 == 0){
                sum += num;
            }
    count++;
    }
    printf("sum of even numbers: %d\n",sum);
    return 0;
}
```

9) Problem Statement 1: Banking System Simulation

Description: Create a simple banking system simulation that allows users to create an account, deposit money, withdraw money, and check their balance. The program should handle
multiple accounts and provide a menu-driven interface.

Requirements:
1. Use appropriate data types for account balance (e.g., float for monetary values) and user input (e.g., int for account numbers).

2. Implement a structure to hold account details (account number, account holder name, balance).

3. Use control statements to navigate through the menu options:
   i. Create Account
   ii. Deposit Money
   iii. Withdraw Money
   iv. Check Balance

4. Ensure that the withdrawal does not exceed the available balance and handle invalid inputs gracefully.

Example Input/Output:

```
 Welcome to the Banking System
 1. Create Account
 2. Deposit Money
 3. Withdraw Money
 4. Check Balance
 5. Exit
 Choose an option: 1
 Enter account holder name: John Doe
 Account created successfully! Account Number: 1001


Choose an option: 2
 Enter account number: 1001
 Enter amount to deposit: 500
 Deposit successful! New Balance: 500.0


Choose an option: 3
 Enter account number: 1001
 Enter amount to withdraw: 200
 Withdrawal successful! New Balance: 300.0


Choose an option: 4
 Enter account number: 1001
 Current Balance: 300.0


Choose an option: 5
 Exiting the system.
```

```c
#include <stdio.h>
#include <string.h>
int main() {
    char accountholdername[20];
    int accNum, accNum1;
    float deposit = 0.0, Withdrawel = 0.0, accountbalance = 0.0;
    int choice;
    printf("welcome to banking system\n");
    while(1){
        printf("\n1. Create Account\n2. Deposit Money\n3. Withdraw Money\n4. Check
Balance\n5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter account holder name: ");
                scanf("%s", &accountholdername);
                printf("Enter account number: ");
                scanf("%d", &accNum);
                accountbalance = 0;
                printf("Account created successfully!\n");

                break;
            case 2:

                float amount;
                printf("Enter account number: ");
                scanf("%d", &accNum1);

                    if (accNum1 == accNum) {
                      printf("Enter amount to deposit: ");
                      scanf("%f", &amount);
                    if (amount > 0) {
                      accountbalance += amount;
                      printf("Deposit successful! New balance: %.2f\n", accountbalance);
                     } else {
                      printf("Invalid amount.\n");
            }
                 break;
            case 3:

                float amount;
                printf("Enter account number: ");
                scanf("%d", &accNum1);

                if (accNum1 == accNum) {
                printf("Enter amount to withdraw: ");
                scanf("%f", &amount);
                if (amount > 0 && amount <= accountbalance) {
                accountbalance -= amount;
                printf("Withdrawal successful! New balance: %.2f\n", accountbalance);
                } else {
                printf(" insufficient amount.\n");
                }
                break;
            case 4:
```

```
            printf("Enter account number: ");
            scanf("%d", &accNum1);
            if (accNum1 == accNum) {
            printf("current Balance: %.2f\n", accountbalance);}
            break;
         case 5:
            printf("Exiting the system.\n");
            return 0;
         default:
            printf("Invalid choice. Please try again.\n");
      }

   return 0;
}}}}
```

10) Problem Statement 4: Weather Data Analysis

Description: Write a program that collects daily temperature data for a month and analyzes it to find the average temperature, the highest temperature, the lowest temperature, and how many days were above average.

Requirements:

1. Use appropriate data types (float for temperatures and int for days).
2. Store temperature data in an array.
3. Use control statements to calculate:
    i. Average Temperature of the month.
    ii. Highest Temperature recorded.
    iii. Lowest Temperature recorded.
    iv. Count of days with temperatures above average.
4. Handle cases where no data is entered.

Example Input/Output:

```
Enter temperatures for each day of the month (30 days):
Day 1 temperature: 72.5
Day 2 temperature: 68.0
...
Day 30 temperature: 75.0
```

```
Average Temperature of Month: XX.X
 Highest Temperature Recorded: YY.Y
 Lowest Temperature Recorded: ZZ.Z
 Number of Days Above Average Temperature: N
```

```
#include<stdio.h>
int main(){
int count=0;
float temp[20], highest=0.0, lowest=0.0, total=0.0, average=0.0;

printf("enter the temperatures \n");

for(int i=0;i<10;i++){
    printf("day %d temperature: \n",i+1);
    scanf("%f",&temp[i]);
     }
     for(int i=0; i<10; i++){
        if(temp[i]>highest){
            highest = temp[i];

        }

     }
     lowest = temp[0];

for(int i=0;i<10;i++){
    if(temp[i]<lowest){
        lowest = temp[i];

    }

}

for(int i=0;i<10;i++){
    total=total+temp[i];
    }

average=total/10.0;

for(int i=0;i<10;i++){
    if(temp[i]>average){
        count++;

    }

}
printf("highest temperature: %f \n", highest);
printf("lowest temperature: %f \n", lowest);
printf("average temperature: %f \n", average);
printf("number of days with temperature above average: %f\n",count);
}
```

11) Problem Statement : Inventory Management System

Description: Create an inventory management system that allows users to manage products in a store. Users should be able to add new products, update existing product quantities, delete products, and view inventory details.

Requirements:
1. Use appropriate data types for product details (e.g., char arrays for product names, int for quantities, float for prices).

2. Implement a structure to hold product information.

3. Use control statements for menu-driven operations:
    i. Add Product
    ii. Update Product Quantity
    iii. Delete Product
    iv. View All Products in Inventory

4. Ensure that the program handles invalid inputs and displays appropriate error messages.

Example Input/Output:

```
Inventory Management System
 1. Add Product
 2. Update Product Quantity
 3. Delete Product
 4. View All Products in Inventory
 5. Exit


Choose an option: 1
 Enter product name: Widget A
 Enter product quantity: 50
 Enter product price: 19.99


Choose an option: 4
 Product Name: Widget A, Quantity: 50, Price: $19.99


Choose an option: 5
 Exiting the system.
```

```
#include <stdio.h>
#include <string.h>
int main() {
   char name[20];
   int quantity;
   float price;
   int choice;
   printf(" inventory system\n");
   while(1){
      printf("\n1. Add product\n2. update product quantity\n3. Delete product\n4. view all
products in inventory\n5. Exit\n");
      printf("Enter your choice: ");
      scanf("%d", &choice);

      switch (choice) {
         case 1:
            printf("Enter product name: \n");
            scanf("%s", &name);
            printf("Enter product quantity: \n");
            scanf("%d", &quantity);
            printf("enter product price: \n");
            scanf("%f", &price);
            break;
         case 2:
            printf("Enter product name: \n");
            scanf("%s", &name);
            printf("update product quantity: \n");
            scanf("%d", &quantity);
            break;
        case 3:
            printf("Enter product to be deleted: \n");
            scanf("%s",&name);
            name[0] ='\0';
            price = 0;
            quantity = 0;
            printf("product is deleted \n");
            break;
        case 4:
            printf("Product Name: %s, Quantity: %d, Price: %f",name,quantity,price);
            break;
        case 5:
            printf("Exiting the system.\n");
            return 0;
        default:
            printf("Invalid choice. Please try again.\n");
      }
   }
   return 0;
}
```