DAY 6

1. WAP to display the multiplication table up to the given limit.

```
#include <stdio.h>
int main()
{
   int n;
   printf("Enter the limit: ");
   scanf("%d", &n);
   int i = 1;
   while (i <= n)
   {
      int j = 1;
      while (j <= 10)
      {
         printf("%d * %d = %d\n", i, j, i * j);
         j++;
      }
      printf("\n");
      i++;
   }
   return 0;
}
```

2. WAP to print the mentioned pattern.

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

```
#include <stdio.h>
int main()
{
  int n, i=1, j=1;
  printf("Enter the number of row: ");
```

```
  scanf("%d", &n);

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

3. WAP to print the mentioned pattern.

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

```
#include <stdio.h>
int main()
{
  int n, i=1, j=1, k=1;
  printf("Enter the number of row: ");
  scanf("%d", &n);

  while(i <= n)
  {
     j=1, k=1;
     while(j <= n-i)
     {
        printf(" ");
        j++;
     }
     while(k <= i)
     {
        printf("* ");
        k++;
     }
     i++;
     printf("\n");
  }
}
```

4. WAP to display the multiplication table up to the given limit (using do-while).

```
#include <stdio.h>

int main()
{
  int n;
   printf("Enter the limit: ");
   scanf("%d", &n);
   int i = 1;
   do
   {
      int j = 1;
      do
      {
         printf("%d * %d = %d\n", i, j, i * j);
         j++;
      }while (j <= 10);
      printf("\n");
      i++;
   }while(i <= n);
   return 0;

}
```

5. WAP to calculate the sum of n natural numbers (using for loop).

```
#include <stdio.h>

int main()
{
  int n, sum=0;
   printf("Enter the natural number: ");
   scanf("%d", &n);

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

6. WAP to reverse a number using for loop.

```
#include <stdio.h>

int main()
{
  int num, rem=0, rev=0;
   printf("Enter the number: ");
   scanf("%d", &num);
```

```
  for( ; num != 0 ; num/=10)
  {
     rem = num % 10;
     rev = rev * 10 +rem;
  }
  printf("The reversed number is %d", rev);
   return 0;
}
```

7. WAP to print the Fibonacci series up to a given limit.

```
#include <stdio.h>
int main()
{
  int n, first=0, second=1, next=0;
   printf("Enter the number: ");
   scanf("%d", &n);

  for(int i=0; next<=n ; i++)
  {
    printf("%d ",next);
    first=second;
    second=next;
    next=first+second;
  }

   return 0;
}
```

8. WAP to print pascals triangle.

```
#include <stdio.h>
int main()
{
  int n;
  printf("Enter the number of rows: ");
  scanf("%d", &n);

  for(int i = 1; i <= n; i++)
  {
     for(int j = 1; j <= n - i; j++)
     {
        printf(" ");
     }

     int a =1;
     for(int k=1; k<=i; k++)
     {
        printf("%d ", a);
        a=a*(i-k)/k;
     }
     printf("\n");
```

```
  }
  return 0;
}
```

9. This is a guessing game.

Requirements

- In this challenge, you are going to create a "Guess the Number" C program
- Your program will generate a random number from 0 to 20
- You will then ask the user to guess it
- User should only be able to enter numbers from 0-20
- The program will indicate to the user if each guess is too high or too low
- The player wins the game if they can guess the number within five tries

Sample Output

I have chosen a number between 0 and 20 which you must guess
You have 5 tries left.
Enter a guess: 12
Sorry, 12 is wrong. My number is less than that.
You have 4 tries left.

Enter a guess: 8
Sorry, 8 is wrong. My number is less than that.

You have 3 tries left.
Enter a guess: 4
Sorry, 4 is wrong. My number is less than that.

You have 2 tries left.
Enter a guess: 2

Congratulations. You guessed it!

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
int generate_random_number(int lower, int upper)
{
   return (rand() % (upper - lower + 1)) + lower;
}
int main()
{
   int try=5;
   int num;

   // Seed the random number generator
   srand(time(0));

   // Generate a random number between 1 and 10
```

```
 int random_no = generate_random_number(1, 20);
 printf("The random number is -> %d\n",random_no);

do
{
   printf("You have %d tries left.\nEnter a guess: ", try);
   scanf("%d", &num);

   if(num == random_no)
   {
      printf("Congratulations. You guessed it!");
      break;
   }
   else if(!(num>=0 && num<=20) )
   {
       printf("Invalid number !!\nEnter a value between 0 – 20\n\n");
   }
   else
   {
      if(num > random_no)
      {
         printf("Sorry, %d is wrong. My number is less than that\n\n",num);
      }
      else
      {
         printf("Sorry, %d is wrong. My number is greater than that\n\n",num);
      }
   }

   try--;

}while(try != 0);
if(try == 0)
{
 printf("Your chances are over!!\n");
}
}
```

10. Problem Statement: Filter Even Numbers with Continue
    Description: Write a C program that prompts the user to enter a series of integers (up to 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.
    Requirements:
    1. Prompt the user for up to 20 integers.
    2. Use a loop to read each integer.
    3. If an integer is negative, use continue to skip adding it to the sum.
    4. If an integer is even, add it to a running total sum.
    5. After all inputs, display the total sum of even numbers.

Example Input/Output:

Enter up to 20 integers (enter 1 to stop):

4

7

3

2

8

-5

10

-1

Sum of even numbers: 24

```
#include <stdio.h>
int main()
{
   int num, i, sum=0;
   printf("Enter up to 20 integers (enter -1 to stop):\n");
   for(i=0; i<20; i++)
   {
      scanf("%d",&num);
      if(num == -1)
      {
         break;
      }
      else if(num % 2 != 0 ||  num < -1)
      {
        continue;
      }
       sum+=num;

   }

   printf("Sum of even numbers: %d",sum);
}
```

11. 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.

```
#include <stdio.h>

int main()

{
```

```
int choice = 0, acc_no = 1001, read_acc_no;
char name[50];
float balance = 0, deposit = 0, withdraw;
do
{
   printf("\nWelcome to the Banking System\n");
   printf("1. Create Account\n");
   printf("2. Deposit Money\n");
   printf("3. Withdraw Money\n");
   printf("4. Check Balance\n");
   printf("5. Exit\n");
   printf("Choose an option: ");
   scanf("%d", &choice);

   switch(choice)
   {
      case 1:
      {
         printf("Enter account holder name: ");
         scanf(" %[^\n]", name);
         printf("Account created successfully! Account Number: 1001\n");
      }
      break;
      case 2:
      {
         printf("Enter account number: ");
         scanf("%d", &read_acc_no);
         if(acc_no == read_acc_no)
         {
            printf("Enter the amount to deposit: ");
            scanf("%f", &deposit);
            balance += deposit;
```

```
        printf("Deposit successful! New Balance: %.2f\n", balance);
    }
    else
    {
        printf("Account number %d is not found!\n", read_acc_no);
    }
}
break;
case 3:
{
    printf("Enter account number: ");
    scanf("%d", &read_acc_no);
    if(acc_no == read_acc_no)
    {
        printf("Enter the amount to withdraw: ");
        scanf("%f", &withdraw);
        if(withdraw <= balance)
        {
            balance -= withdraw;
            printf("Withdrawal successful! New Balance: %.2f\n", balance);
        }
        else
        {
            printf("Insufficient balance!\n");
        }
    }
    else
    {
        printf("Account number %d is not found!\n", read_acc_no);
    }
}
break;
```

```
            case 4:
            {
                printf("Enter account number: ");
                scanf("%d", &read_acc_no);
                if(acc_no == read_acc_no)
                {
                    printf("Current balance: %.2f\n", balance);
                }
                else
                {
                    printf("Account number %d is not found!\n", read_acc_no);
                }
            }
            break;

            case 5:
                printf("Exiting the Banking System.\n");
                break;

            default:
                printf("Invalid option! Please try again.\n");
        }
    } while(choice != 5);

    return 0;
}
```

12. 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 no_av=0;

   float temp[30], sum=0, largest, smallest, avg;

   printf("Enter temperatures for each day of the month (30 days): ");

   for(int i=0; i<30; i++)

   {

      scanf("%f", &temp[i]);

      if(i == 1)

      {

         largest = temp[i];

         smallest = temp[i];

      }

      else if(temp[i] > largest)

      {

         largest = temp[i];

      }

      else if(temp[i] < smallest)
```

```
    {
        smallest = temp[i];
    }

    sum+=temp[i];
  }
  avg=sum/30;
  for(int i=1; i<=30; i++)
  {
    if(temp[i] > avg)
    {
      no_av++;
    }
  }
  printf("Average Temperature of Month: %.2f\n", avg);
  printf("Highest Temperature Recorded: %.2f\n", largest);
  printf("Lowest Temperature Recorded: %.2f\n", smallest);
  printf("Number of days above average temperature: %d",no_av);

}
```

13.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>

int main()

{

   int choice=0, pro_quantity=0;

   char pro_name[50];

   float pro_price;

   do

   {

      printf("\nInventory Management System\n");

      printf("1. Add Product\n");

      printf("2. Update Product Quantity\n");

      printf("3. Delete Product\n");

      printf("4. View All Products in Inventory\n");

      printf("5. Exit\n");

      printf("Choose an option: ");

      scanf("%d",&choice);

      switch(choice)

      {

         case 1:

         {

            printf("Enter product name: ");
```

```
    getchar();
    scanf("%[^\n]", pro_name);
    printf("Enter product quantity: ");
    scanf("%d", &pro_quantity);
    printf("Enter product price: ");
    scanf("%f", &pro_price);
}
break;
case 2:
{
    printf("Enter new product quantity: ");
    scanf("%d", &pro_quantity);
}
break;
case 3:
{
    pro_quantity=0;
    pro_price=0;
    printf("Product delected successfully!!\n");
}
break;
case 4:
{
    if(pro_quantity != 0)
    {
        printf("Product name: %s,", pro_name);
        printf("Quantity: %d,", pro_quantity);
        printf("Price: $%.2f.\n", pro_price);
    }
    else
    {
        printf("No product available.\n");
```

```
            }
        }
        break;
        case 5:
        {
            printf("Exiting system.\n");
        }
        break;
        default:
        printf("Enter valid option!!\n");
    }
  }while(choice != 5);

}
```