While loop

```
#include <stdio.h>

int main()
{
    char a=1;
    while(a<=10);{
    printf("a=%d",a);
    a++;
}
    return 0;
}
```

Output

NOP

Qn.Multiplication table

```
#include <stdio.h>

int main()
{
    char i=1,j;
    while(i<=10){
        j=1;
        while(j<=10){
         printf("%d *%d=%d",i,j,i*j);
            j++;}
    printf("\n");
    i++;
}
    return 0;
}
```

Output

```
1 *1=11 *2=21 *3=31 *4=41 *5=51 *6=61 *7=71 *8=81 *9=91 *10=10
2 *1=22 *2=42 *3=62 *4=82 *5=102 *6=122 *7=142 *8=162 *9=182 *10=20
3 *1=33 *2=63 *3=93 *4=123 *5=153 *6=183 *7=213 *8=243 *9=273 *10=30
4 *1=44 *2=84 *3=124 *4=164 *5=204 *6=244 *7=284 *8=324 *9=364 *10=40
5 *1=55 *2=105 *3=155 *4=205 *5=255 *6=305 *7=355 *8=405 *9=455 *10=50
6 *1=66 *2=126 *3=186 *4=246 *5=306 *6=366 *7=426 *8=486 *9=546 *10=60
7 *1=77 *2=147 *3=217 *4=287 *5=357 *6=427 *7=497 *8=567 *9=637 *10=70
8 *1=88 *2=168 *3=248 *4=328 *5=408 *6=488 *7=568 *8=648 *9=728 *10=80
9 *1=99 *2=189 *3=279 *4=369 *5=459 *6=549 *7=639 *8=729 *9=819 *10=90
10 *1=1010 *2=2010 *3=3010 *4=4010 *5=5010 *6=6010 *7=7010 *8=8010 *9=9010 *10=100
```

Qn.print the pattern using while loop

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

```
#include <stdio.h>

int main()
{
    char i=1,j;
    while(i<=5){
        j=1;
        while(j<=i){
         printf("*");
            j++;}
    printf("\n");
    i++;
}
    return 0;
}
```

Output

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

Qn. WAP to print a full pyramid using while loop

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

    while (i <= rows) {
        int j = 1,k=1;
        while (k <= rows - i) {
            printf(" ");
            k++;
        }
        while (j <= i) {
            printf("* ");
            j++;

        }
         printf("\n");
            i++;
    }
    return 0;
}
```

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

DO WHILE LOOP

Qn.WAP to print the numbers between 1 to 10 using do while

```
#include <stdio.h>
int main() {
    int i = 1;
    do{
        printf("%d \n",i);
        i++;
    }
    while (i <=10);
    return 0;
}
```

```
1
2
3
4
5
6
7
8
9
10
```

Qn.multiplication table using do while

```
#include <stdio.h>
int main() {
    char i = 1, j;

    do {
        j = 1;

        do {
            printf("%d * %d = %d\n", i, j, i * j);
            j++;
        } while (j <= 10);
        i++;
    } while (i <= 10);
    return 0;
}
```

Output

```
1 * 1 = 1

1 * 2 = 2

1 * 3 = 3

1 * 4 = 4

1 * 5 = 5

1 * 6 = 6

1 * 7 = 7

1 * 8 = 8

1 * 9 = 9

1 * 10 = 10

2 * 1 = 2

2 * 2 = 4

2 * 3 = 6

2 * 4 = 8

2 * 5 = 10

2 * 6 = 12

2 * 7 = 14

2 * 8 = 16

2 * 9 = 18

2 * 10 = 20

3 * 1 = 3

3 * 2 = 6

3 * 3 = 9

3 * 4 = 12

3 * 5 = 15

3 * 6 = 18
```

3 * 7 = 21

3 * 8 = 24

3 * 9 = 27

3 * 10 = 30

4 * 1 = 4

4 * 2 = 8

4 * 3 = 12

4 * 4 = 16

4 * 5 = 20

4 * 6 = 24

4 * 7 = 28

4 * 8 = 32

4 * 9 = 36

4 * 10 = 40

5 * 1 = 5

5 * 2 = 10

5 * 3 = 15

5 * 4 = 20

5 * 5 = 25

5 * 6 = 30

5 * 7 = 35

5 * 8 = 40

5 * 9 = 45

5 * 10 = 50

6 * 1 = 6

6 * 2 = 12

6 * 3 = 18

6 * 4 = 24

6 * 5 = 30

6 * 6 = 36

6 * 7 = 42

6 * 8 = 48

6 * 9 = 54

6 * 10 = 60

7 * 1 = 7

7 * 2 = 14

7 * 3 = 21

7 * 4 = 28

7 * 5 = 35

7 * 6 = 42

7 * 7 = 49

7 * 8 = 56

7 * 9 = 63

7 * 10 = 70

8 * 1 = 8

8 * 2 = 16

8 * 3 = 24

8 * 4 = 32

8 * 5 = 40

8 * 6 = 48

8 * 7 = 56

8 * 8 = 64

8 * 9 = 72

8 * 10 = 80

9 * 1 = 9

9 * 2 = 18

9 * 3 = 27

9 * 4 = 36

```
9 * 5 = 45

9 * 6 = 54

9 * 7 = 63

9 * 8 = 72

9 * 9 = 81

9 * 10 = 90

10 * 1 = 10

10 * 2 = 20

10 * 3 = 30

10 * 4 = 40

10 * 5 = 50

10 * 6 = 60

10 * 7 = 70

10 * 8 = 80

10 * 9 = 90

10 * 10 = 100
```

FOR LOOP

```
#include <stdio.h>
int main() {
   int i;
   for(i=1;i<=10;i++)
   {
      printf("%d \n",i);
   }
   return 0;
}
```

```
 1
2
3
4
5
6
7
8
9
10
```

Qn.WAP to calculate the sum of first n natural numbers

```
#include <stdio.h>
int main() {
   int i,n,sum=0;
   printf("Enter the number:");
   scanf("%d",&n);
   for(i=0;i<n;i++)
   {
      sum=sum+i;
   }
   printf("%d \n",sum);
   return 0;
}
```

```
 Enter the number:5
10
```

Qn.WAP to reverse a number using for loop

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

```
int main() {
    int i,n,rev=0,rem;
    printf("Enter the number:");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        rem=n%10;
        rev=rev*10+rem;
        n=n/10;
    }
    printf("%d \n",rev);
    return 0;
}
```

```
Enter the number:35
53
```

Qn.WAP to print Fibonocci series upto N terms usinf for loop

```
#include <stdio.h>
int main() {
    int n1=0,n2=1,temp,N;
    printf("Enter the number:");
    scanf("%d",&N);
    for(int i=0;i<N;i++)
    {
        printf("%d ", n1);
        temp=n1+n2;
        n1=n2;
        n2=temp;

    }

    return 0;
}
```

```
 Enter the number:10
0 1 1 2 3 5 8 13 21 34
```

Qn.Implement pascals triangle print it till 8 rows using for loop

```
#include <stdio.h>

int main() {
    int rows = 8;
    int coeff = 1;



    for (int i = 0; i < rows; i++) {

        for (int j = 0; j < rows - i - 1; j++) {
            printf(" ");
        }

        for (int j = 0; j <= i; j++) {
            if (j == 0) {
                coeff = 1;
            } else {
```

```
            coeff = coeff * (i - j + 1) / j;
          }
          printf("%d ", coeff);
        }
        printf("\n");
    }

    return 0;
}
```

```
      1
     1 1
    1 2 1
   1 3 3 1
  1 4 6 4 1
 1 5 10 10 5 1
```

Infinite for loop

```
#include <stdio.h>


int main(){
    printf("Program is halted");
    for(;;);
    return 0;
}
```

qn. 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

Same Output

This is a guessing game. 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 main() {

   int number, guess, attempts = 5;
   srand(time(0));
   number = rand() % 21;

   printf("This is a guessing game. I have chosen a number between 0 and 20 which you must
guess.\n");

   for (int i = attempts; i > 0; i--) {
      printf("You have %d tries left. Enter a guess: ", i);
      scanf("%d", &guess);
      if (guess < 0 || guess > 20) {
         printf("Please enter a number between 0 and 20.\n");
         i++;
         continue;
      }

      if (guess == number) {
            printf("Congratulations. You guessed it!\n");
            return 0;
        } else if (guess > number) {
            printf("Sorry, %d is wrong. My number is less than that.\n", guess);
        } else {
            printf("Sorry, %d is wrong. My number is greater than that.\n", guess);
        }
    }

    printf("Sorry, you've used all your tries. The number was %d.\n", number);
    return 0;
}
```

```
This is a guessing game. I have chosen a number between 0 and 20 which you must guess.

You have 5 tries left. Enter a guess: 10

Sorry, 10 is wrong. My number is less than that.

You have 4 tries left. Enter a guess: 15

Sorry, 15 is wrong. My number is less than that.

You have 3 tries left. Enter a guess: 9

Sorry, 9 is wrong. My number is less than that.

You have 2 tries left. Enter a guess: 7

Sorry, 7 is wrong. My number is less than that.

You have 1 tries left. Enter a guess: 18

Sorry, 18 is wrong. My number is less than that.

Sorry, you've used all your tries. The number was 4.
```

BREAK

```
#include <stdio.h>


int main()

{
    int i;
    for(i=0;i<=10;i++){
        if(i==5){
            break;
        }
    printf("%d\n",i);
}
    return 0;
}
```

```
0
1
2
3
4
```

Continue

```
#include <stdio.h>

int main()
{
    int i;
    for(i=0;i<=10;i++){
        if(i==5){
            continue;
        }
    printf("%d\n",i);
}
```

```
    return 0;
}
```

```
0
1
2
3
4
6
7
8
9
10
```

Qn. 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
```

```c
#include <stdio.h>

int main() {
    int sum = 0;
    int num, count = 0;

    printf("Enter up to 20 integers (enter 1 to stop):\n");

    while (count < 20) {
        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;
}
```

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

4

7

-3

2

8

-5

10

1

Sum of even numbers: 24
```

Qn. 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>

#include <string.h>

#define MAX_ACCOUNTS 100


int accountNumbers[MAX_ACCOUNTS];

char accountHolderNames[MAX_ACCOUNTS][50];

float balances[MAX_ACCOUNTS];

int accountCount = 0;


void createAccount() {

  if (accountCount >= MAX_ACCOUNTS) {

    printf("Cannot create more accounts. Limit reached.\n");

    return;

  }


  int accountNumber = 1000 + accountCount;


  printf("Enter account holder name: ");

  getchar();

  fgets(accountHolderNames[accountCount], 50, stdin);

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

  balances[accountCount] = 0.0;


  accountNumbers[accountCount] = accountNumber;

  accountCount++;
```

```
    printf("Account created successfully! Account Number: %d\n", accountNumber);
}


int findAccountIndex(int accountNumber) {
    for (int i = 0; i < accountCount; i++) {
        if (accountNumbers[i] == accountNumber) {
            return i;
        }
    }
    return -1;
}


void depositMoney() {
    int accountNumber;
    float amount;

    printf("Enter account number: ");
    scanf("%d", &accountNumber);

    int index = findAccountIndex(accountNumber);
    if (index == -1) {
        printf("Account not found.\n");
        return;
    }

    printf("Enter amount to deposit: ");
    scanf("%f", &amount);

    if (amount <= 0) {
        printf("Invalid deposit amount.\n");
        return;
    }

    balances[index] += amount;
    printf("Deposit successful! New Balance: %.2f\n", balances[index]);
}


void withdrawMoney() {
    int accountNumber;
    float amount;

    printf("Enter account number: ");
    scanf("%d", &accountNumber);

    int index = findAccountIndex(accountNumber);
    if (index == -1) {
        printf("Account not found.\n");
        return;
    }

    printf("Enter amount to withdraw: ");
    scanf("%f", &amount);

    if (amount <= 0) {
        printf("Invalid withdrawal amount.\n");
        return;
    }

    if (amount > balances[index]) {
        printf("Insufficient balance for withdrawal.\n");
    } else {
        balances[index] -= amount;
        printf("Withdrawal successful! New Balance: %.2f\n", balances[index]);
    }
}



void checkBalance() {
    int accountNumber;

    printf("Enter account number: ");
    scanf("%d", &accountNumber);

    int index = findAccountIndex(accountNumber);
    if (index == -1) {
        printf("Account not found.\n");
        return;
    }

    printf("Current Balance: %.2f\n", balances[index]);
}
void displayMenu() {
    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: ");
}

int main() {
    int choice;

    while (1) {
        displayMenu();
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                createAccount();
                break;
            case 2:
                depositMoney();
                break;
            case 3:
                withdrawMoney();
                break;
            case 4:
                checkBalance();
                break;
            case 5:
                printf("Exiting the system.\n");
                return 0;
            default:
                printf("Invalid option! Please choose again.\n");
        }
    }

    return 0;
}
```

```
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: Kannan
Account created successfully! Account Number: 1000

Welcome to the Banking System
1. Create Account
2. Deposit Money
3. Withdraw Money
4. Check Balance
5. Exit
Choose an option: 2
Enter account number: 1000
Enter amount to deposit: 5000000
```

```
Deposit successful! New Balance: 5000000.00

Welcome to the Banking System
1. Create Account
2. Deposit Money
3. Withdraw Money
4. Check Balance
5. Exit
Choose an option: 3
Enter account number: 1000
Enter amount to withdraw: 30000
Withdrawal successful! New Balance: 4970000.00

Welcome to the Banking System
1. Create Account
2. Deposit Money
3. Withdraw Money
4. Check Balance
5. Exit
Choose an option: 4
Enter account number: 1000
Current Balance: 4970000.00

Welcome to the Banking System
1. Create Account
2. Deposit Money
3. Withdraw Money
4. Check Balance
5. Exit
```

```
Choose an option: 5

Exiting the system.
```

Qn. 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 days = 30;
 float temperatures[30], sum = 0.0, average;
 float highest, lowest;
 int daysAboveAverage = 0;
 for (int i = 0; i < days; i++) {
 printf("Enter temperature for Day %d: ", i + 1);
 scanf("%f", &temperatures[i]);
 sum += temperatures[i];
if (i == 0) {
 highest = temperatures[i];
 lowest = temperatures[i];
 } else {
 if (temperatures[i] > highest)
 highest = temperatures[i];
 if (temperatures[i] < lowest)
 lowest = temperatures[i];
 }
 }
 average = sum / days;
 for (int i = 0; i < days; i++) {
 if (temperatures[i] > average) {
 daysAboveAverage++;
 }
 }
 printf("\nAverage Temperature of Month: %.1f\n", average);
 printf("Highest Temperature Recorded: %.1f\n", highest);
 printf("Lowest Temperature Recorded: %.1f\n", lowest);
 printf("Number of Days Above Average Temperature: %d\n", daysAboveAverage);
 return 0;
}
```

OUTPUT

```
Enter temperature for Day 1: 25
Enter temperature for Day 2: 45
Enter temperature for Day 3: 15
Enter temperature for Day 4: 40
Enter temperature for Day 5: 62
Enter temperature for Day 6: 3
Enter temperature for Day 7: 41
Enter temperature for Day 8: 22
Enter temperature for Day 9: 24
Enter temperature for Day 10: 130
Enter temperature for Day 11: 120
Enter temperature for Day 12: 452
Enter temperature for Day 13: -23
Enter temperature for Day 14: -45
Enter temperature for Day 15: -62
Enter temperature for Day 16: -42
Enter temperature for Day 17: -75
Enter temperature for Day 18: 36
Enter temperature for Day 19: 100
Enter temperature for Day 20: 110
Enter temperature for Day 21: 160
Enter temperature for Day 22: 41
Enter temperature for Day 23: 162
Enter temperature for Day 24: 753
Enter temperature for Day 25: 32
Enter temperature for Day 26: 23
Enter temperature for Day 27: 14
Enter temperature for Day 28: 101
Enter temperature for Day 29: 105
Enter temperature for Day 30: 109
Average Temperature of Month: 82.6
Highest Temperature Recorded: 753.0
Lowest Temperature Recorded: -75.0
Number of Days Above Average Temperature: 11
```

Qn. 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.

```
#include <stdio.h>
#include <string.h>
#define MAX_PRODUCTS 100
int main() {
 char productNames[MAX_PRODUCTS][50];
 int productQuantities[MAX_PRODUCTS];
 float productPrices[MAX_PRODUCTS];
 int productCount = 0;
 int choice;
 while (1) {
 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);
 if (choice == 1) {
 if (productCount >= MAX_PRODUCTS) {
 printf("Inventory is full. Cannot add more products\n");
 } else {
 printf("Enter product name: ");
 scanf("%s", productNames[productCount]);
 printf("Enter product quantity: ");
 scanf("%d", &productQuantities[productCount]);
 printf("Enter product price: ");
 scanf("%f", &productPrices[productCount]);
 productCount++;
 printf("Product added successfully\n");
 }
 } else if (choice == 2) {
```

```
char name[50];
int newQuantity, found = 0;
printf("Enter product name to update quantity: ");
scanf("%s", name);
for (int i = 0; i < productCount; i++) {
if (strcmp(productNames[i], name) == 0) {
printf("Enter new quantity: ");
scanf("%d", &newQuantity);
productQuantities[i] = newQuantity;
printf("Quantity updated successfully\n");
found = 1;
break;
}
}
if (!found) {
printf("Product not found\n");
}
} else if (choice == 3) {
char name[50];
int found = 0;
printf("Enter product name to delete: ");
scanf("%s", name);
for (int i = 0; i < productCount; i++) {
if (strcmp(productNames[i], name) == 0) {
for (int j = i; j < productCount - 1; j++) {
strcpy(productNames[j], productNames[j + 1]);
productQuantities[j] = productQuantities[j + 1];
productPrices[j] = productPrices[j + 1];
}
productCount--;
printf("Product deleted successfully\n");
found = 1;
break;
}
}
if (!found) {
printf("Product not found\n");
}
} else if (choice == 4) {
if (productCount == 0) {
printf("Inventory is empty\n");
} else {
printf("Inventory Details:\n");
for (int i = 0; i < productCount; i++) {
printf("Product Name: %s Quantity: %d Price: %.2f\n",
productNames[i], productQuantities[i], productPrices[i]);
}
}
```

```
 } else if (choice == 5) {
 printf("Exiting the system\n");
 break;
 } else {
 printf("Invalid choice. Please try again\n");
 }
}
 return 0;
}
```

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
Product added successfully
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 B
Enter product quantity: 30
Enter product price: 29.99
Product added successfully
Inventory Management System
1. Add Product
2. Update Product Quantity
3. Delete Product
4. View All Products in Inventory
5. Exit
Choose an option: 4
Inventory Details:
Product Name: Widget A Quantity: 50 Price: 19.99
Product Name: Widget B Quantity: 30 Price: 29.99
Inventory Management System
1. Add Product
2. Update Product Quantity
3. Delete Product
4. View All Products in Inventory
5. Exit
```

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