

## sumN3.c

```
1  #include <stdio.h>
2
3  int main() {
4      int n, i = 1, sum = 0;
5
6      printf("Enter a positive integer: ");
7      scanf("%d", &n);
8
9      if (n <= 0) {
10         printf("Please enter a positive integer.\n");
11         return 1;
12     }
13
14     // using while loop
15     while (i <= n) {
16         sum = sum + i;
17         i++;
18     }
19
20     // using do-while loop
21     /*
22     do {
23         sum = sum + i;
24         i++;
25     } while (i <= n);
26     */
27
28     printf("Sum of the first %d natural numbers is %d\n", n, sum);
29
30     return 0;
31 }
32
```

## sumN2.c

```
1  #include <stdio.h>
2
3  int main(){
4
5      int n, sum;
6
7      printf("Enter a positive integer: ");
8      scanf("%d", &n);
9
10     if (n <= 0) {
11         printf("Please enter a positive integer.\n");
12         return 1;
13     }
14
15     sum = n * (n + 1) / 2;
16
17     printf("Sum of the first %d natural numbers is %d\n", n, sum);
18
19     return 0;
20 }
```

## sumN.c

```
1  #include <stdio.h>
2
3  int main() {
4      int n, i, sum = 0;
5
6      printf("Enter a positive integer: ");
7      scanf("%d", &n);
8
9      // Input validation for natural numbers (which start from 1)
10     if (n <= 0) {
11         printf("Please enter a positive integer.\n");
12         return 1; // Exit with an error code
13     }
14
15     for (i = 1; i <= n; i++) {
16         sum = sum + i;
17     }
18
19     printf("Sum of the first %d natural numbers is %d\n", n, sum);
20
21     return 0;
22 }
23
```

## examples/squareRoot.c

```
1  #include <stdio.h>
2  #include <math.h>
3
4  // Function to calculate the square root of a number
5
6  int main() {
7      double number, result;
8
9      // Prompt the user for input
10     printf("Enter a number to find its square root: ");
11     scanf("%lf", &number);
12
13     // Check if the number is non-negative
14     if (number < 0) {
15         printf("Error: Cannot compute the square root of a negative number.\n");
16         return 1;
17     }
18
19     // Calculate the square root
20     result = sqrt(number);
21
22     // Display the result
23     printf("The square root of %.3f is %.3f\n", number, result);
24
25     return 0;
26 }
```

**solidSquare.c**

```
1  #include <stdio.h>
2
3  // solid square
4
5  int main()
6  {
7
8      int rows;
9
10     printf("Enter number of rows: ");
11     scanf("%d",&rows);
12
13
14     // print pattern
15     for (int i = 1; i <= rows; i++)
16     // outer loop for rows
17     {
18         for (int j = 1; j <= rows; j++)
19         // inner loop for columns
20         {
21             printf("* ");
22         }
23         printf("\n");
24     }
25
26     return 0;
27 }
28
29
30 /**
31  * Example Output:
32  * Enter number of rows: 5
33  * * * * *
34  * * * * *
35  * * * * *
36  * * * * *
37  * * * * *
38  */
```

**solidRectangle.c**

```
1  #include <stdio.h>
2
3  // Solid Rectangle Program
4
5  int main(){
6
7      // define variables for rows and columns
8      int rows,col;
9      // get user input
10     printf("Enter number of rows: ");
11     scanf("%d",&rows);
12     printf("Enter number of columns: ");
13     scanf("%d",&col);
14
15     // print pattern
16     for (int i = 0; i < rows; i++)
17     // outer loop for rows
18     {
19         for (int j = 0; j < col; j++)
20         // inner loop for columns
21         {
22             printf("* ");
23         }
24         printf("\n");
25     }
26
27     /**
28      * Alternative method
29      */
30
31     /*
32     // print solid rectangle
33     for (int i = 1; i <= rows; i++)
34     // outer loop for rows
35     {
36         for (int j = 1; j <= col; j++)
37         // inner loop for columns
38         {
39             printf("* ");
40         }
41         printf("\n");
42     }
43     */
44
45
46
47
48     return 0;
49 }
50
51
```

```
52  /**
53  * Example Output:
54  * Enter number of rows: 4
55  * Enter number of columns: 5
56  * * * * *
57  * * * * *
58  * * * * *
59  * * * * *
60  */
```

**rightHalfPyramid.c**

```
1  #include <stdio.h>
2
3  // Right Half Pyramid Pattern
4
5  int main()
6  {
7
8      int rows;
9
10     printf("Enter number of rows: ");
11     scanf("%d",&rows);
12
13
14     // print pattern
15     for (int i = 1; i <= rows; i++)
16     // outer loop for rows
17     {
18         for (int j = 1; j <= i; j++)
19         // inner loop for columns
20         {
21             printf("* ");
22         }
23         printf("\n");
24     }
25
26     return 0;
27 }
28
```



**leftHalfPyramid.c**

```
1  #include <stdio.h>
2
3  // Left Half Pyramid Pattern
4
5  int main()
6  {
7      int rows;
8
9      printf("Enter number of rows: ");
10     scanf("%d",&rows);
11
12     // print pattern
13     for (int i = 1; i <= rows; i++)
14         // outer loop for rows
15         {
16
17             // logic for spaces
18             for (int space = 1; space <= rows - i; space++)
19                 {
20                     printf(" ");
21                 }
22
23             for (int j = 1; j <= i; j++)
24                 // inner loop for columns
25                 {
26                     printf("* ");
27                 }
28             printf("\n");
29         }
30
31     return 0;
32 }
33
34
35 /*
36  * Example Output:
37  * Enter number of rows: 5
38  *           *
39  *         * *
40  *       * * *
41  *     * * * *
42  * * * * *
43  */
```

**InvertedRHP.c**

```
1  #include <stdio.h>
2
3  // Inverted Right Half Pyramid Pattern
4
5  int main()
6  {
7
8      int rows;
9
10     printf("Enter number of rows: ");
11     scanf("%d",&rows);
12
13
14     // print pattern
15     for (int i = 1; i <= rows; i++)
16     // outer loop for rows
17     {
18         for (int j = 1; j <= rows - i + 1; j++)
19         // inner loop for columns
20         {
21             printf("* ");
22         }
23         printf("\n");
24     }
25
26     return 0;
27 }
28
29 /**
30  * Example Output:
31  * Enter number of rows: 5
32  * + + + + +
33  * + + + +
34  * + + +
35  * + +
36  * +
37  */
```

**ILHP.c**

```
1  #include <stdio.h>
2
3  // Inverted Left Half Pyramid Pattern
4
5  int main()
6  {
7      int rows;
8
9      printf("Enter number of rows: ");
10     scanf("%d",&rows);
11
12     // print pattern
13     for (int i = 1; i <= rows; i++)
14     // outer loop for rows
15     {
16         // logic for spaces
17         for (int space = 1; space <= i - 1; space++)
18         {
19             printf("  ");
20         }
21
22         for (int j = 1; j <= rows - i + 1; j++)
23         // inner loop for columns
24         {
25             printf("* ");
26         }
27         printf("\n");
28     }
29
30     return 0;
31 }
32
33
34 /*
35  * Example Output:
36  * Enter number of rows: 5
37  * * * * *
38  *  * * *
39  *    * *
40  *      *
41  */
42
```

## fact2.c

```
1  #include <stdio.h>
2
3  int main() {
4      int n, i = 1;
5      int factorial = 1;
6      printf("Enter a non-negative integer: ");
7      scanf("%d", &n);
8
9      // Check if the number is negative
10     if (n < 0) {
11         printf("Error: Factorial is not defined for negative numbers.\n");
12     } else {
13         do {
14             factorial = factorial * i;
15             i++;
16         } while (i <= n);
17         printf("Factorial of %d is %d \n", n, factorial);
18     }
19
20     return 0;
21 }
22
```

## fact.c

```
1  #include <stdio.h>
2
3  int main() {
4      int n, i = 1;
5      int factorial = 1;
6      printf("Enter a non-negative integer: ");
7      scanf("%d", &n);
8
9      // Check if the number is negative
10     if (n < 0) {
11         printf("Error: Factorial is not defined for negative numbers.\n");
12     } else {
13         // Calculate the factorial using a while loop
14         while (i <= n) {
15             factorial = factorial * i; // Multiply current factorial by i
16             i++; // Increment the counter
17         }
18         printf("Factorial of %d is %d \n", n, factorial);
19     }
20
21     return 0;
22 }
23
```

## checkPrime.c

```
1  #include <stdio.h>
2  #include <math.h>
3
4  // check if a number is prime
5
6  int main(){
7
8      int num, i, isPrime = 1;
9
10     printf("Enter a positive integer: ");
11     scanf("%d", &num);
12
13     if (num <= 1) {
14         isPrime = 0; // numbers less than or equal to 1 are not prime
15     } else {
16         for (i = 2; i <= sqrt(num); i++) {
17             if (num % i == 0) {
18                 isPrime = 0; // found a divisor, not prime
19                 break;
20             }
21         }
22     }
23
24     if (isPrime) {
25         printf("%d is a prime number.\n", num);
26     } else {
27         printf("%d is not a prime number.\n", num);
28     }
29
30     return 0;
31 }
```

## p2.c

```
1  #include <stdio.h>
2  /*
3   * Print the following pattern for given number of rows.
4   * Enter number of rows: 5
5   * 1
6   * 2 2
7   * 3 3 3
8   * 4 4 4 4
9   * 5 5 5 5 5
10  */
11
12  int main()
13  {
14
15      int rows;
16
17      printf("Enter number of rows: ");
18      scanf("%d",&rows);
19
20
21      // print pattern
22      for (int i = 1; i <= rows; i++)
23          // outer loop for rows
24          {
25              for (int j = 1; j <= i; j++)
26                  // inner loop for columns
27                  {
28                      //printf("* ");
29                      printf("%d ",i);
30                  }
31              printf("\n");
32          }
33
34      return 0;
35  }
36
```

**p1.c**

```
1  #include <stdio.h>
2  /*
3   * Print the following pattern for given number of rows.
4   * Enter number of rows: 5
5   * 1
6   * 1 2
7   * 1 2 3
8   * 1 2 3 4
9   * 1 2 3 4 5
10  */
11
12  int main()
13  {
14
15      int rows;
16
17      printf("Enter number of rows: ");
18      scanf("%d",&rows);
19
20
21      // print pattern
22      for (int i = 1; i <= rows; i++)
23          // outer loop for rows
24          {
25              for (int j = 1; j <= i; j++)
26                  // inner loop for columns
27                  {
28                      //printf("* ");
29                      printf("%d ",j);
30                  }
31              printf("\n");
32          }
33
34      return 0;
35  }
36
```