

PROGRAM 1: Program to calculate sum, difference, product, quotient and remainder of two numbers

```
1 /* Program to calculate sum, difference, product, quotient
2  * and remainder of two numbers
3  */
4
5 # include <stdio.h>
6
7 int main()
8 {
9     int num_1 = 0,
10        num_2 = 0,
11        sum = 0,
12        difference = 0,
13        product = 0,
14        quotient = 0,
15        remainder = 0;
16
17     printf ("Enter num_1 = ");
18     scanf ("%d", &num_1);
19     printf ("Enter num_2 = ");
20     scanf ("%d", &num_2);
21
22     sum = num_1 + num_2;
23     difference = num_1 - num_2;
24     product = num_1 * num_2;
25     quotient = num_1 / num_2;
26     remainder = num_1 % num_2;
27
28     printf ("\n");
29     printf ("Sum = %d\n", sum);
30     printf ("Difference = %d\n", difference);
31     printf ("Product = %d\n", product);
32     printf ("Quotient = %d\n", quotient);
33     printf ("Remainder = %d\n", remainder);
34     return 0;
35 }
36
```

```
~/C > main ± xecut operators.c
Enter num_1 = 25
Enter num_2 = 12

Sum = 37
Difference = 13
Product = 300
Quotient = 2
Remainder = 1

~/C > main ± xecut operators.c
Enter num_1 = 56
Enter num_2 = 4

Sum = 60
Difference = 52
Product = 224
Quotient = 14
Remainder = 0

~/C > main ±
```

PROGRAM 2: Program to demonstrate pre and post increment operators in C

```
1 /* Program to demonstrate pre-post increment operators
2  */
3
4 # include <stdio.h>
5
6 int main()
7 {
8     int x = 5, y = 7, result = 0;
9
10    printf ("Before calculation:\n"
11           " x = %d\n"
12           " y = %d\n", x, y);
13
14    // at this stage result = 5 + 8 and x = 6, y = 8
15    result = x++ + ++y;
16
17    printf ("\nResult = %d\n", result);
18
19    printf ("After calculation:\n"
20           " x = %d\n"
21           " y = %d\n", x, y);
22
23    return 0;
24 }
```

```
11:10 >_ 69%
~/C > main ± xecut pre_post_inc.c
Before calculation:
x = 5
y = 7

Result = 13

After calculation:
x = 6
y = 8

~/C > main ±
```

PROGRAM 3: Program to demonstrate type casting in C

```
1 /* Program to demonstrate type casting
2  */
3
4 # include <stdio.h>
5
6 int main()
7 {
8     int x = 8574, y = 100;
9
10    /* NOTE: this casts x to float and then does the division
11     * If (float) isn't used, the result will only be 85.
12     */
13    float result = (float) x / y;
14
15    printf ("Result = %lf\n", result);
16    return 0;
17 }
18
```

```
1 /* Program to demonstrate type casting using accuracy format string
2  */
3
4 # include <stdio.h>
5
6 int main()
7 {
8     int x = 8574, y = 100;
9
10    /* NOTE: this casts x to float and then does the division
11     * If (float) isn't used, the result will only be 85.
12     */
13    float result = (float) x / y;
14
15    printf ("Result = %0.2lf\n", result);
16    return 0;
17 }
18
```

```
10:59 >_ HD LTE 70%
~/C > xcut type_casting_with_accuracy.c
Result = 85.74
~/C > xcut type_casting.c
Result = 85.739998
~/C >
```

PROGRAM 4: WACP for grading marks obtained by a student using conditional statements

```
1 /* Program for grading marks obtained by a student
2  * using conditional statements.
3  */
4
5 #include <stdio.h>
6
7 int main()
8 {
9     int student_marks = 0;
10
11     printf ("Enter marks (out of 100): ");
12     scanf ("%d", &student_marks);
13
14     if (student_marks < 0) {
15         printf ("error: marks entered subceeds 0\n");
16         return 0;
17     } else if (student_marks > 100) {
18         printf ("error: marks entered exceeds 100\n");
19         return 0;
20     }
21
22     printf ("Grade: ");
23     if (student_marks <= 40) {
24         printf ("F\n");
25     } else if (student_marks > 40 && student_marks <= 55) {
26         printf ("E\n");
27     } else if (student_marks > 55 && student_marks <= 70) {
28         printf ("D\n");
29     } else if (student_marks > 70 && student_marks <= 85) {
30         printf ("C\n");
31     } else if (student_marks > 85 && student_marks <= 95) {
32         printf ("B\n");
33     } else if (student_marks > 95 && student_marks <= 100) {
34         printf ("A\n");
35     }
36     return 0;
37 }
38
```

```
10:59 >_ HD LTE 70%
~/C > xcut grading_system.c
Enter marks (out of 100): 96
Grade: A
~/C > xcut grading_system.c
Enter marks (out of 100): 85
Grade: C
~/C > xcut grading_system.c
Enter marks (out of 100): 87
Grade: B
~/C > xcut grading_system.c
Enter marks (out of 100): 45
Grade: E
~/C > xcut grading_system.c
Enter marks (out of 100): 40
Grade: F
~/C >
```

EXECUTION BASH SCRIPT (xecute):

```
1 #!/bin/bash
2
3 file="$1"
4 if ! [ "$file" = "" ]
5 then
6     exec_name=$(basename "$file" ".c")
7     gcc -g -Wall "$file" -o "$exec_name"
8     ./"$exec_name"
9     rm "$exec_name"
10 else
11     echo "error: no file name passed"
12 fi
13
```