arithmetic_operator.c

```c
#include <stdio.h>
void main()
{
int a = 9,b = 4, c;
c = a+b;
printf("a+b = %d \n",c);
c = a-b;
printf("a-b = %d \n",c);
c = a*b;
printf("a*b = %d \n",c);
c=a/b;
printf("a/b = %d \n",c);
c=a%b;
printf("Remainder when a divided by b = %d \n",c);
}

/*
* Output: a=9, b=4, c=0
*/

/*
* Note : If we use void main instead of int main, return 0 statement is not required.
* However, using int main is considered a better practice as it indicates that the program has executed successfully.
*/
```

arithmetic_operator_2.c

```c
#include <stdio.h>
// C program to demonstrate arithmetic operators
int main()
{
int a = 9,b = 4, c;
c = a+b;
printf("a+b = %d \n",c);
c = a-b;
printf("a-b = %d \n",c);
c = a*b;
printf("a*b = %d \n",c);
c=a/b;
printf("a/b = %d \n",c);
c=a%b;
printf("Remainder when a divided by b = %d \n",c);
return 0;
}

/*
* Output:
a+b = 13
a-b = 5
a*b = 36
a/b = 2
Remainder when a divided by b = 1
*/


```

relational_operator.c

```c
#include <stdio.h>
// C program to demonstrate relational operators
int main()
{
int a = 5, b = 5, c = 10;
printf("%d == %d = %d \n", a, b, a == b); // true
printf("%d == %d = %d \n", a, c, a == c); // false
printf("%d > %d = %d \n", a, b, a > b); //false
printf("%d > %d = %d \n", a, c, a > c); //false
printf("%d < %d = %d \n", a, b, a < b); //false
printf("%d < %d = %d \n", a, c, a < c); //true
printf("%d != %d = %d \n", a, b, a != b); //false
printf("%d != %d = %d \n", a, c, a != c); //true
printf("%d >= %d = %d \n", a, b, a >= b); //true
printf("%d >= %d = %d \n", a, c, a >= c); //false
printf("%d <= %d = %d \n", a, b, a <= b); //true
printf("%d <= %d = %d \n", a, c, a <= c); //true
return 0;
}

/*
* Output:
5 == 5 = 1
5 == 10 = 0
5 > 5 = 0
5 > 10 = 0
5 < 5 = 0
5 < 10 = 1
5 != 5 = 0
5 != 10 = 1
5 >= 5 = 1
5 >= 10 = 0
5 <= 5 = 1
5 <= 10 = 1
*/
```

logical_operator.c

```c
#include <stdio.h>

// C program to demonstrate logical operators

int main (){
    int a = 5, b = 5, c = 10, result;
    result = (a = b) && (c > b);
    printf("(a = b) && (c > b) equals to %d \n", result);
    result = (a = b) && (c < b);
    printf("(a = b) && (c < b) equals to %d \n", result);
    result = (a = b) || (c < b);
    printf("(a = b) || (c < b) equals to %d \n", result);
    result = (a != b) || (c < b);
    printf("(a != b) || (c < b) equals to %d \n", result);
    result = !(a != b);
    printf("!(a == b) equals to %d \n", result);
    result = !(a == b);
    printf("!(a == b) equals to %d \n", result);
    return 0;
}

/*****************************
* Output: a=b=5, c=10
*****************************
(a = b) && (c > b) equals to 1
(a = b) && (c < b) equals to 0
(a = b) || (c < b) equals to 1
(a != b) || (c < b) equals to 0
!(a == b) equals to 1
!(a == b) equals to 0
*/
```

assignment_operator.c

```c
#include <stdio.h>

// C program to demonstrate assignment operators

int main (){

    // variable declaration
    int a,b,c;

   // variable initialization
   a = 5;
   b = 10;
   c = 15;

   // using assignment operators
   a += 5; // equivalent to a = a + 5
   b -= 5; // equivalent to b = b - 5
   c *= 2; // equivalent to c = c * 2

   // printing the results
   printf("After assignment operations:\n");
   printf("a = %d\n", a);
   printf("b = %d\n", b);
   printf("c = %d\n", c);

   return 0;
}

/*****************************
* Output: a=5, b=10, c=15
*****************************
After assignment operations:
a = 10
b = 5
c = 30
*/
```

assignment_operator_2.c

```c
// C Program to demonstrate the working of assignment operators
#include <stdio.h>
int main()
{
int a = 5, c;
c = a;
printf("c = %d \n", c);
c += a; // c = c+a
printf("c = %d \n", c);
c -= a; // c = c-a
printf("c = %d \n", c);
c *= a; // c = c*a
printf("c = %d \n", c);
c /= a; // c = c/a
printf("c = %d \n", c);
c %= a; // c = c%a
printf("c = %d \n", c); // Remainder of c divided by a
return 0;
}

/*****************************
* Output: a=5, c=5
*****************************
c = 5
c = 10
c = 5
c = 25
c = 5
c = 0
*/
```