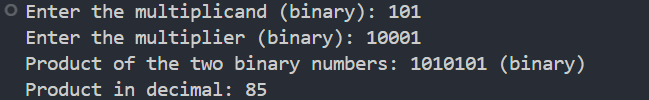
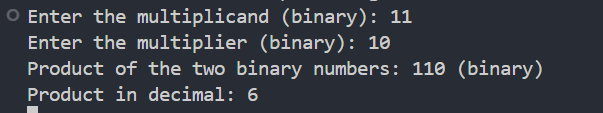
**SOURCE CODE**

1. #include <stdio.h>
2. #include <conio.h>
4. // Function **to** perform binary addition
5. int binaryAddition(int a, int b) {
6. int carry = 0, result = 0, bit = 1;
7. while (a != 0 || b != 0) {
8. int bit\_a = a % 10;
9. int bit\_b = b % 10;
11. // Add **the** bits along **with** **the** carry
12. int sum = bit\_a + bit\_b + carry;
14. // Update **the** result
15. result += (sum % 2) \* bit;
17. // Calculate **the** carry for **the** next bit
18. carry = sum / 2;
20. // Move **to** **the** next bit
21. a /= 10;
22. b /= 10;
23. bit \*= 10;
24. }
26. // Add **the** carry **if** exists
27. result += carry \* bit;
28. **return** result;
29. }
31. // Function **to** perform logical shift left
32. int logicalShiftLeft(int num) {
33. **return** num \* 10; // Equivalent **to** shifting left by 1 position
34. }
36. // Function **to** perform logical shift right
37. int logicalShiftRight(int num) {
38. **return** num / 10; // Equivalent **to** shifting right by 1 position
39. }
41. // Function **to** perform Booth's multiplication algorithm
42. int boothMultiply(int multiplicand, int multiplier) {
43. int accumulator = 0;
44. int bitMask = 1;
46. // Iterate over **each** bit **of** **the** multiplier
47. while (multiplier != 0) {
48. // Step 2: Test Y0; **if** **it** **is** 1, add content **of** X **to** **the** accumulator A
49. **if** (multiplier % 10 == 1) {
50. accumulator = binaryAddition(accumulator, multiplicand);
51. }
53. // Step 3: Logical Shift **the** content **of** X left one position **and** content **of** Y right one position
54. multiplicand = logicalShiftLeft(multiplicand);
55. multiplier = logicalShiftRight(multiplier);
57. // Move **the** bit mask **to** **the** next bit
58. bitMask \*= 10;
59. }
61. **return** accumulator;
62. }
64. // Function **to** convert binary number **to** decimal
65. int binaryToDecimal(int binary) {
66. int decimal = 0, base = 1;
67. while (binary != 0) {
68. int lastDigit = binary % 10;
69. decimal += lastDigit \* base;
70. binary /= 10;
71. base \*= 2;
72. }
73. **return** decimal;
74. }
76. int main() {
77. int multiplicand, multiplier;
78. printf("Enter the multiplicand (binary): ");
79. scanf("%d", &multiplicand);
80. printf("Enter the multiplier (binary): ");
81. scanf("%d", &multiplier);
83. // Step 1: Clear **the** accumulator (sum)
84. int product = boothMultiply(multiplicand, multiplier);
86. printf("Product of the two binary numbers: %d (binary)\n", product);
87. printf("Product in decimal: %d\n", binaryToDecimal(product));
89. getch();
90. **return** 0;
91. }

**OUTPUT**





*Fig: Output of Code*