1. Write a program to convert binary numbers to decimal numbers using a for loop.

```
#include<bits/stdc++.h>

using namespace std;

int main() {
    int binaryNumber;

    cout << "Enter a binary number: ";
    cin >> binaryNumber;

         int decimal = 0;
    int base = 1;

    while (binary > 0) {
        int lastDigit = binary % 10;
        decimal += lastDigit * base;
        binary /= 10;
        base *= 2;
    }

    cout << "The decimal equivalent is: " << decimal <<
endl;

    return 0;
}
```

2. Write a program to convert decimal numbers to binary numbers using a for loop.

```
#include <iostream>

using namespace std;

int main() {
    int decimalNumber;
    int binaryNumber = 0;
    int base = 1;

    cout << "Enter a decimal number: ";
    cin >> decimalNumber;

    while (decimalNumber > 0) {
        int remainder = decimalNumber % 2;
        binaryNumber += remainder * base;
        decimalNumber /= 2;
        base *= 10;
    }

    cout << "The binary equivalent is: " << binaryNumber
<< endl;

    return 0;
}
```

3. Write a program to convert decimal numbers to Octal numbers.

```
#include <iostream>

using namespace std;

int main() {
    int decimalNumber;
    int octalNumber = 0;
    int base = 1;

    cout << "Enter a decimal number: ";
    cin >> decimalNumber;

    while (decimalNumber > 0) {
        int remainder = decimalNumber % 8;
        octalNumber += remainder * base;
        decimalNumber /= 8;
        base *= 10;
    }

    cout << "The octal equivalent is: " << octalNumber <<
endl;

    return 0;
}
```

4. Write a program to convert Octal numbers to decimal numbers.

```cpp
  #include <iostream>

using namespace std;

int main() {
    int octalNumber, decimalNumber = 0;
    int base = 1; // Represents the current octal digit's
positional value

    cout << "Enter an octal number: ";
    cin >> octalNumber;

    while (octalNumber > 0) {
        int lastDigit = octalNumber % 10; // Get the last octal
digit
        decimalNumber += lastDigit * base; // Add the decimal
equivalent to the result
        octalNumber /= 10; // Move to the next octal digit
        base *= 8; // Increase the positional value for the next
digit
    }

    cout << "The decimal equivalent is: " << decimalNumber <<
endl;

    return 0;
}
```

5. Write a program to convert binary to Octal numbers

```
#include <iostream>

using namespace std;

int main() {
    int binaryNumber;
    cout << "Enter a binary number: ";
    cin >> binaryNumber;

    int octalNumber = 0, decimalNumber = 0;
    int base = 1; // Represents the current binary digit's
positional value

    while (binaryNumber > 0) {
        int lastDigit = binaryNumber % 10; // Get the last
binary digit
        decimalNumber += lastDigit * base; // Convert to
decimal
        binaryNumber /= 10; // Move to the next binary digit
        base *= 2; // Increase the positional value for the
next digit
    }

    int octalDigit, i = 1;
    while (decimalNumber != 0) {
        octalDigit = decimalNumber % 8; // Get the remainder
when dividing by 8
        octalNumber += octalDigit * i;  // Build the octal
number
```

```
        decimalNumber /= 8; // Reduce the decimal number
        i *= 10; // Increase the positional value for the next
octal digit
    }

    cout << "The octal equivalent is: " << octalNumber << endl;

    return 0;
}
```

6 . Write a program to convert Octal numbers to binary numbers

```
#include <iostream>

using namespace std;

int main() {
    int octalNumber, binaryNumber = 0;
    int base = 1; // Represents the current octal digit's
positional value

    cout << "Enter an octal number (as an integer): ";
    cin >> octalNumber;

    int tempOctal = octalNumber;

    while (tempOctal != 0) {
        int octalDigit = tempOctal % 10; // Get the last octal
digit
        int binaryDigit = 0;
```

```
        int binaryPlaceValue = 1;

        // Convert each octal digit to a 3-digit binary number
        while (octalDigit > 0) {
            binaryDigit += (octalDigit % 2) * binaryPlaceValue;
            octalDigit /= 2;
            binaryPlaceValue *= 10;
        }

        binaryNumber += binaryDigit * base; // Add the binary
equivalent to the result
        tempOctal /= 10; // Move to the next octal digit
        base *= 1000; // Increase the positional value for the
next digit
    }

    cout << "The binary equivalent is: " << binaryNumber <<
endl;

    return 0;
}
```