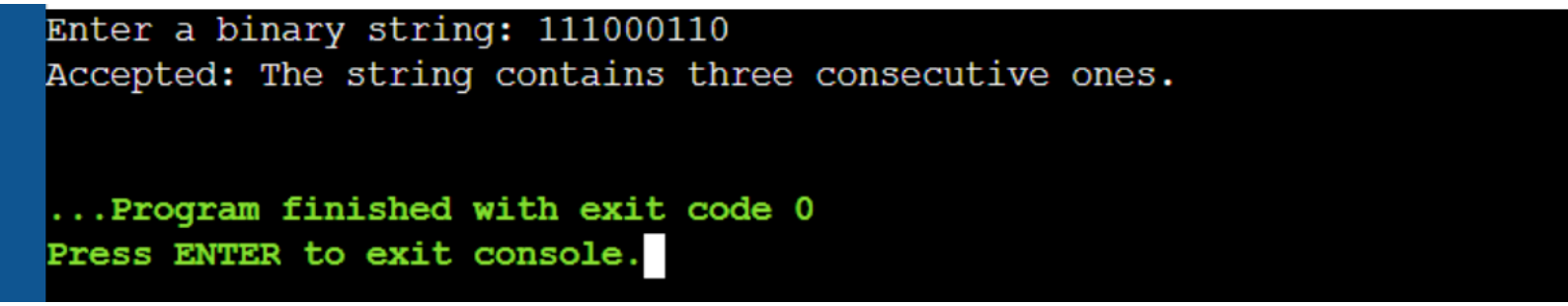
1.Design a program for creating machine that accept three consecutive one .

```
#include <iostream>
#include <string>
using namespace std;
bool acceptThreeConsecutiveOnes(const string& input) {
int state = 0; // Initial state
for (char ch : input) {
if (ch == '1') {
if (state < 3) {
state++;
}
} else {
state = 0; // Reset state if any other character is encountered
}
// Check if we have reached state 3 (three consecutive ones)
if (state == 3) {
return true;
}
}
return false; // Return false if we don't encounter three consecutive
ones
}
int main() {
string input;
cout << "Enter a binary string: ";
cin >> input;
```

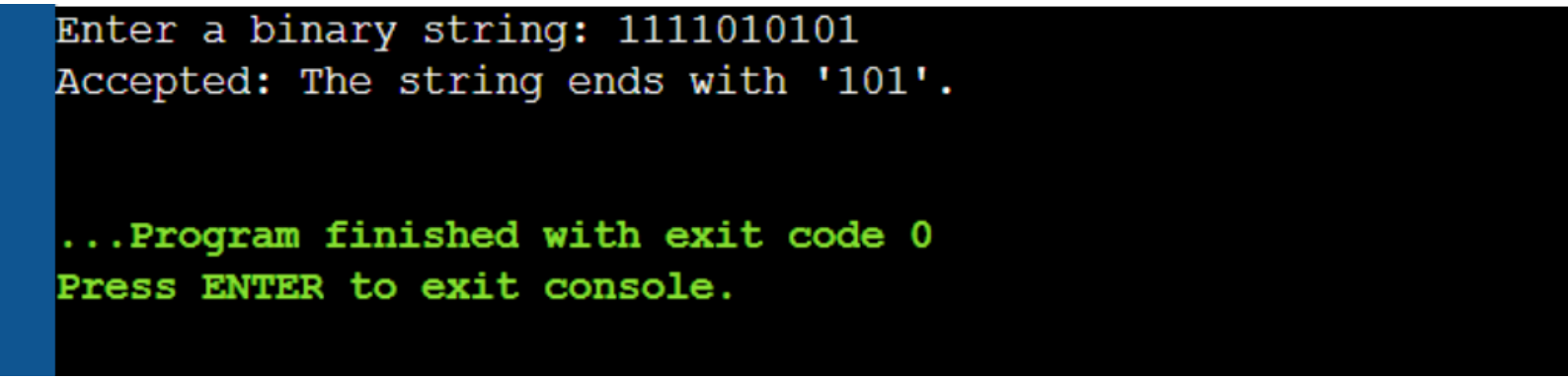
```
if (acceptThreeConsecutiveOnes(input)) {
cout << "Accepted: The string contains three consecutive
ones.\n";
} else {
cout << "Rejected: The string does not contain three consecutive
ones.\n";
}
return 0;
}
```



2. Write a simple program in c++ for creating a machine that accept that string always ending with 101

```
#include <iostream>
#include <string>
using namespace std;
bool acceptStringEndingWith101(const string& input) {
    // Check if the input string is long enough to contain "101" at the end
    if (input.size() < 3) {
        return false; // Too short to end in "101"
    }
    // Compare the last three characters with "101"
```

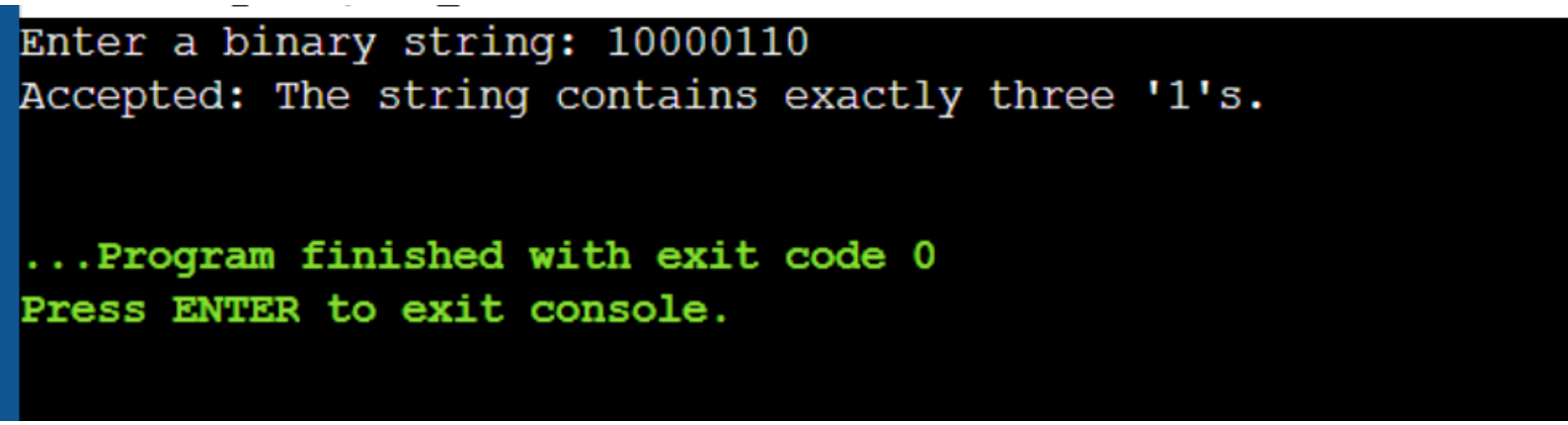
```
    return input.substr(input.size() - 3) == "101";
}
int main() {
    string input;
    cout << "Enter a binary string: ";
    cin >> input;
    if (acceptStringEndingWith101(input)) {
        cout << "Accepted: The string ends with '101'.\n";
    } else {
        cout << "Rejected: The string does not end with '101'.\n";
    }
    return 0;
}
```



3. Design a program for mode 3 machine.

```
#include <iostream>
#include <string>
using namespace std;
bool mode3MachineAccept(const string& input) {
    int countOfOnes = 0;      // Counter for the number of '1's in the input
```

```
    for (char ch : input) {
        if (ch == '1') {
            countOfOnes++;
        } else if (ch != '0') {
            // Invalid character encountered (not '0' or '1')
            return false;
        }
        // If we exceed three '1's, we can stop early
        if (countOfOnes > 3) {
            return false;
        }
    }
    // Accept the string only if exactly three '1's are found
    return countOfOnes == 3;
}
int main() {
    string input;
    cout << "Enter a binary string: ";
    cin >> input;
    if (mode3MachineAccept(input)) {
        cout << "Accepted: The string contains exactly three '1's.\n";
    } else {
        cout << "Rejected: The string does not contain exactly three '1's.\n";
    }
    return 0;
}
```



4. Design a program for accepting decimal number divisible by 2 .

```
#include <iostream>
#include <string>
using namespace std;
bool isDivisibleBy2(const string& binary) {
    // Check if the last character of the binary string is '0'
    return binary.back() == '0';
}
int main() {
    string binaryInput;
    cout << "Enter a binary string: ";
    cin >> binaryInput;
    // Check if the input consists only of '0's and '1's
    for (char ch : binaryInput) {
        if (ch != '0' && ch != '1') {
            cout << "Invalid input: Only '0' and '1' are allowed.\n";
            return 1;
        }
    }
```

```
    if (isDivisibleBy2(binaryInput)) {
        cout << "Accepted: The binary string represents a decimal number divisible by 2.\n";
    } else {
        cout << "Rejected: The binary string does not represent a decimal number divisible by 2.\n";
    }
    return 0;
}
```

```
Enter a binary string: 1110
Accepted: The binary string represents a decimal number divisible by 2.


...Program finished with exit code 0
Press ENTER to exit console.
```

5. Design a program for creating a machine which accept string having equal number of 1 and 0.

```
#include <iostream>
#include <string>
using namespace std;
bool hasEqualOnesAndZeros(const string& input) {
    int countOnes = 0;
    int countZeros = 0;
    // Count the number of '1's and '0's in the input string
    for (char ch : input) {
        if (ch == '1') {
            countOnes++;
        } else if (ch == '0') {
```

```
            countZeros++;
        } else {
            // Invalid character (not '0' or '1')
            cout << "Invalid input: Only '0' and '1' are allowed.\n";
            return false;
        }
    }
    // Check if the counts of '1's and '0's are equal
    return countOnes == countZeros;
}
int main() {
    string input;
    cout << "Enter a binary string: ";
    cin >> input;
    if (hasEqualOnesAndZeros(input)) {
        cout << "Accepted: The string has an equal number of '1's and '0's.\n";
    } else {
        cout << "Rejected: The string does not have an equal number of '1's and '0's.\n";
    }
    return 0;
}
```

```
Enter a binary string: 11110000
Accepted: The string has an equal number of '1's and '0's.


...Program finished with exit code 0
Press ENTER to exit console.
```

6.Design a program for creating a machine which count number of 1s and 0 in a given string.

```
#include <iostream>
#include <string>
using namespace std;
void countOnesAndZeros(const string& input) {
    int countOnes = 0;
    int countZeros = 0;
    // Iterate over each character in the input string
    for (char ch : input) {
        if (ch == '1') {
            countOnes++;
        } else if (ch == '0') {
            countZeros++;
        } else {
            // Invalid character found
            cout << "Invalid input: Only '0' and '1' are allowed.\n";
            return;
        }
    }
    // Display the counts of '1's and '0's
    cout << "Number of '1's: " << countOnes << "\n";
    cout << "Number of '0's: " << countZeros << "\n";
}
int main() {
```

```
    string input;
    cout << "Enter a binary string: ";
    cin >> input;
    countOnesAndZeros(input);
    return 0;
}
```

```
Enter a binary string: 111000001
Number of '1's: 4
Number of '0's: 5


...Program finished with exit code 0
Press ENTER to exit console.
```

7. Design a program to find 2s complement of a given binary number.

```
#include <iostream>
#include <string>
#include <algorithm>
using namespace std;
string findTwosComplement(const string& binary) {
    string invertedBinary = binary;
    // Step 1: Invert all bits (0 becomes 1, 1 becomes 0)
    for (char& ch : invertedBinary) {
        ch = (ch == '0') ? '1' : '0';
    }
    // Step 2: Add 1 to the inverted binary string
    int n = invertedBinary.length();
```

```
    bool carry = true;
    for (int i = n - 1; i >= 0; --i) {
        if (invertedBinary[i] == '1' && carry) {
            invertedBinary[i] = '0';
        } else if (carry) {
            invertedBinary[i] = '1';
            carry = false;
        }
    }
    // If there is still a carry, it means an overflow for fixed-length representation (e.g., 4 bits).
    // Here we assume no overflow.
    return invertedBinary;
}
int main() {
    string binary;
    cout << "Enter a binary number: ";
    cin >> binary;
    // Check if the input is a valid binary string
    for (char ch : binary) {
        if (ch != '0' && ch != '1') {
            cout << "Invalid input: Only '0' and '1' are allowed.\n";
            return 1;
        }
    }
    string twosComplement = findTwosComplement(binary);
    cout << "The two's complement of " << binary << " is: " << twosComplement << "\n";
```

```
    return 0;
}
```

```
Enter a binary number: 11100
The two's complement of 11100 is: 00100


...Program finished with exit code 0
Press ENTER to exit console.
```

8.Design a program which will increment the given binary number by 1.

```
#include <iostream>
#include <string>

using namespace std;
string incrementBinary(const string& binary) {
    string result = binary;
    int n = result.length();
    // Start from the rightmost bit and add 1
    bool carry = true;
    for (int i = n - 1; i >= 0; --i) {
        if (result[i] == '1' && carry) {
            result[i] = '0';  // Set to 0 and keep carry
        } else if (carry) {
            result[i] = '1';  // Set to 1 and stop carrying
            carry = false;
```

```
        }
    }
    // If there's still a carry, it means an overflow (e.g., "111" + 1 = "1000")
    if (carry) {
        result.insert(result.begin(), '1'); // Add '1' at the beginning
    }
    return result;
}
int main() {
    string binary;
    cout << "Enter a binary number: ";
    cin >> binary;
    // Check if the input is a valid binary string
    for (char ch : binary) {
        if (ch != '0' && ch != '1') {
            cout << "Invalid input: Only '0' and '1' are allowed.\n";
            return 1;
        }
    }
    string incrementedBinary = incrementBinary(binary);
    cout << "The binary number after incrementing " << binary << " by 1 is: " <<
incrementedBinary << "\n";
    return 0;
}
```

```
Enter a binary number: 111111011
The binary number after incrementing 111111011 by 1 is: 111111100


...Program finished with exit code 0
Press ENTER to exit console.
```