This C++ program checks whether the given expression has balanced brackets or not. The work process of the program can be described as follows:

1. The program starts by including the necessary header file `bits/stdc++.h`, which includes all the standard C++ libraries.

2. It defines a function `areBracketsBalanced` that takes a string `expr` as an input and returns a boolean value `true` if the brackets are balanced, and `false` otherwise.

3. Inside the `areBracketsBalanced` function:

a. It declares a stack `temp` to hold the brackets encountered during the iteration.

b. It iterates through each character of the `expr` string using a for loop.

c. For each character in the expression:

- If the stack is empty, it means we have encountered an opening bracket, so we push it into the stack.

- If the stack is not empty, it means we have an expression with both an opening and closing bracket:

- If the top of the stack matches the current character, it means we have a complete pair (e.g., (), {}, []), so we pop the top element from the stack.

- If the top of the stack does not match the current character, it means the brackets are not balanced, so we push the current character into the stack.

d. After the loop finishes, we check if the stack is empty. If it is empty, it means all brackets had matching pairs, and the expression is balanced. If not, it means some brackets were not closed properly, and the expression is not balanced.

4. The `main` function is the entry point of the program.

a. It declares a string variable `expr`.

b. It takes the input expression from the user using `cin`.

c. It then calls the `areBracketsBalanced` function with the input expression and stores the result in a boolean variable.

d. Finally, it prints "Balanced" if the expression is balanced (the function returned `true`) and "Not Balanced" otherwise (the function returned `false`).

Overall, the program checks the input expression for balanced brackets using a stack data structure to keep track of the brackets encountered. If all opening brackets have matching closing brackets in the right order, the program determines the expression as balanced; otherwise, it considers it as not balanced.