You qualify for the pie-throwing festival.

The entered age didn't match any of the test ranges, so the program set index to 3 and then printed the corresponding string.

Program Notes

The expression `age > 17 && age < 35` tests for ages between the two values, that is, ages in the range 18–34. The expression `age >= 35 && age < 50` uses the `<=` operator to include 35 in its range, which is 35–49. If the program had used `age > 35 && age < 50`, the value 35 would have been missed by all the tests. When you use range tests, you should check that the ranges don't have holes between them and that they don't overlap. Also, be sure to set up each range correctly; see the note on Range Tests.

The `if else` statement serves to select an array index, which, in turn, identifies a particular string.

Range Tests

Note that each part of a range test should use the AND operator to join two complete relational expressions:

```
if (age > 17 && age < 35)   // OK
```

Don't borrow from mathematics and use the following notation:

```
if (17 < age < 35)          // Don't do this!
```

If you make this error, the compiler won't catch it, for it still is valid C++ syntax. The `<` operator associates from left to right, so the previous expression means the following:

```
if ( (17 < age) < 35)
```

But `17 < age` is either true, or 1, or else false, or 0. In either case, the expression `17 < age` is less than 35, so the entire test is always true!