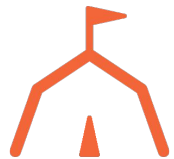
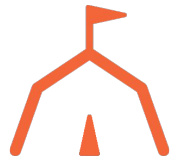
Boolean Expressions



Boolean Expressions

Recall that a boolean expression is one that evaluates to either true or false. We can use these expressions to determine the flow of our control statements.



Boolean Expressions & Control Statements

In programs, we usually need to do one set of code if something is true, and a different set of code if that something isn't true.

This is where the if statement comes in.

Boolean Expressions & Control Statements

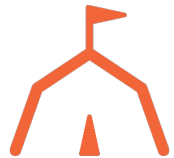
The key is understanding how to test if things are true.

To do that, we use expressions and relational operators to compare things.

Boolean Expressions & Control Statements

Example: The user can enter a number for a radius of a circle so we can calculate area.

But if they enter a negative number, we print out an error message.



Boolean Expressions & Control Statements

Conceptually:

1. User enters a number.
2. If the number is less than 0,

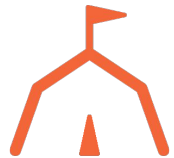
    Print out an error message

    Otherwise

    Calculate the area

Relational Operators

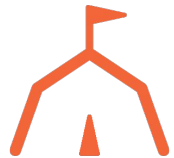
- The six relational operators (listed in following slide) are used to compare operands which are primitive data types.
- Operands may be literals, variables, arithmetic expressions, or keywords.



Relational Operators

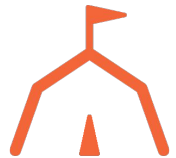
Example:

```
if (value < 0)
{
    Console.WriteLine("Please do not use negative numbers.");
}
else
{
    double area = Math.PI * value * value;
    Console.WriteLine(area);
}
```



Relational Operators

Operator	Name
==	Equality
!=	Inequality
>	Greater Than
<	Less Than
>=	Greater Than or Equal
<=	Less Than or Equal



Example On Boolean Expressions

```
discountPercent == 2.3 // equal to a numeric literal
letter == 'y' // equal to a char literal
isValid == false // equal to the false value
subtotal != 0 // not equal to a numeric literal
years > 0 // greater than a numeric literal
i < months // less than a variable
subtotal >= 500 // greater than or equal to a numeric literal
quantity <= reorderPoint // less than or equal to a variable
isValid // isValid is equal to true
!isValid // isValid is equal to false
```

Logical Operators

Operator	Name
&&	And
||	Or
!	Not

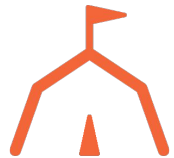
Selection Statements

- If Statement
- Switch Statement
- Ternary Operator

The If Statement

An if statement identifies which statement to run based on the value of a Boolean expression. Example:

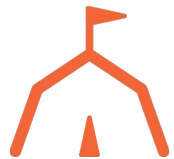
```
bool condition = true;
if (condition)
        {
            Console.WriteLine("Variable is set to true.");
        }
else
        {
            Console.WriteLine("Variable is set to false.");
        }
```



The Switch Statement

The switch statement is a control statement that selects a switch section to execute from a list of candidates. Example:

```
int caseSwitch = 1;
switch (caseSwitch)
  { case 1: Console.WriteLine("Case 1");
    break;
    case 2: Console.WriteLine("Case 2");
    break;
    default: Console.WriteLine("Default case");
    break;}
```



The Conditional Operator

The Conditional Operator may be used in provide a conditional value.

```
(condition) ? value-if-true : value-if-false
```

```
(num % 2 == 0) ? “Even” : “Odd”
```

Also known as the “ternary operator)

(Only operator with 3 parameters)