

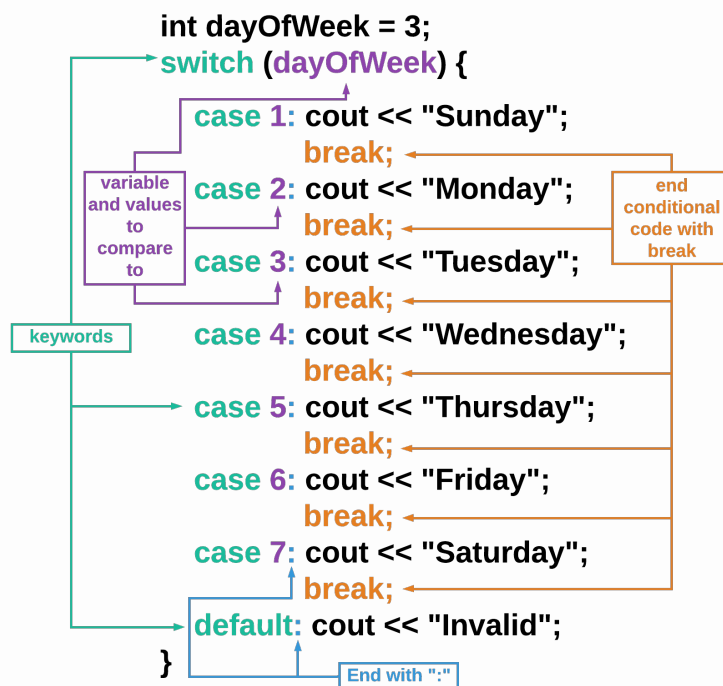
Learning Objectives: Switch Case Statement

- Describe the `switch case` syntax
- Identify when to apply `switch case` statements instead of nested `if-else`

Switch Case Statement Syntax

Swith Case Statement Syntax

The `switch case` statement is a way to make a decision with multiple possible outcomes. Instead of nesting or sequencing many `if` statements, C++ allows you to write the following:



[.guides/img/SwitchCase](#)

Here are the rules for writing a switch case statement:

- Start with `switch` followed by the variable that is going to be tested in parentheses `()`.
- All of the cases are surrounded by a set of curly braces `{}`.
- Each case is followed by a *numerical* value and a colon `:`.
- After each `:`, write the code that should run if the variable is equal to that case's value.
- After each section of code per case, include `break;`.
- As the very last case, use `default:` to specify what should happen if none of the above cases are true.

```
int dayOfWeek = 3;

switch (dayOfWeek) {

    case 1: cout << "Sunday"; //only prints if dayOfWeek == 1
            break;
    case 2: cout << "Monday"; //only prints if dayOfWeek == 2
            break;
    case 3: cout << "Tuesday"; //only prints if dayOfWeek == 3
            break;
    case 4: cout << "Wednesday"; //only prints if dayOfWeek == 4
            break;
    case 5: cout << "Thursday"; //only prints if dayOfWeek == 5
            break;
    case 6: cout << "Friday"; //only prints if dayOfWeek == 6
            break;
    case 7: cout << "Saturday"; //only prints if dayOfWeek == 7
            break;
    default: cout << "Invalid"; //only prints if none of the above
             are true

}
```

Code Visualizer

challenge

What happens if you:

- Assign dayOfWeek to 5?
- Assign dayOfWeek to 0?
- Assign dayOfWeek to 3 and remove all of the break; statements?

Code Visualizer

Switch Case vs. If Else

Switch Case vs. Else If

C++ allows you to use either `switch case` or a series of `else if` statements to handle decisions with multiple outcomes. There are a couple of reasons why you would use one method over the other.

#1: Else If is used for *ranges* of values - Switch Case is for *specific* values

`switch case` can only check for equality (e.g. `num == 5`), so if you need to check for a range of values (e.g. `num > 50 && num <= 60`), use `else If` instead.

```
int grade = 62;
int letterGrade = grade / 10;
switch (letterGrade) {
    case 10: case 9: cout << "A";
                break;
    case 8: cout << "B";
                break;
    case 7: cout << "C";
                break;
    case 6: cout << "D";
                break;
    default: cout << "F";
}

int grade = 62;
if (grade < 60) {
    cout << "F"; }
else if (grade < 70) {
    cout << "D"; }
else if (grade < 80) {
    cout << "C"; }
else if (grade < 90) {
    cout << "B"; }
else if (grade <= 100) {
    cout << "A"; }
```

[.guides/img/SwitchCaseElseIf](#)

▼ What is case 10: case 9:?

Sometimes, the code for multiple cases is the same. Instead of repeating code, you can list multiple cases before the code. Here is another example:

```

int month = 2;
int year = 2000;
int numDays = 0;

switch (month) {
    case 1: case 3: case 5:
    case 7: case 8: case 10:
    case 12:
        numDays = 31;
        break;
    case 4: case 6:
    case 9: case 11:
        numDays = 30;
        break;
    case 2:
        if ((year % 4 == 0) &&
            ! (year % 100 == 0)) ||
            (year % 400 == 0))
            numDays = 29;
        else
            numDays = 28;
        break;
    default:
        cout << "Invalid month.";
        break;
}
cout << "Number of Days = " << numDays << endl;

```

In some cases, as shown above, you can exploit patterns to force ranges into a switch case, but frequently that is not possible and it also makes the code less readable. For example, above, the user has to realize that letterGrade is using integer division to retrieve the tens place of the original grade.

```
int grade = 62;
int letterGrade = grade / 10;
switch (letterGrade) {
    case 10: case 9: cout << "A";
        break;
    case 8: cout << "B";
        break;
    case 7: cout << "C";
        break;
    case 6: cout << "D";
        break;
    default: cout << "F";
}
```

[Code Visualizer](#)

#2: Else If is used for handling multiple variables

switch case can only compare against values - not variables. For example, if you wanted to compare the inputted day of the week with the current day of the week, you would need to use else if. switch case can handle values (dayOfWeek == "Sunday") but not variables (dayOfWeek == today).

#3: Else If is used for compound conditionals

To check multiple conditions, an else if is needed.

Below is an example of a multiple choice grader using switch case:

```
int studentAnswer = 3;
string feedback1 = "This answer is wrong because...";
string feedback2 = "This answer is correct! You know this
because...";
string feedback3 = "This answer is wrong. While the first part
is correct...";
string feedback;

int correctAnswer = 2;
int points = 0;

switch (studentAnswer) {
    case 1: feedback = feedback1; break;
    case 2: feedback = feedback2; break;
    case 3: feedback = feedback3; break;
    default: feedback = "Invalid answer choice";
}

cout << feedback << endl;
```

challenge

Switch Case to Else If

- Change the switch case statements above into else if statements.
- Add a check to see if studentAnswer == correctAnswer.
- If the student's answer is correct, increment (++) the points variable.
- Print out the student's earned points at the end of the program using the points variable.

[Code Visualizer](#)

▼ Sample solution

```
if (studentAnswer == 1) {  
    cout << feedback1 << endl;  
}  
else if (studentAnswer == 2) {  
    cout << feedback2 << endl;  
}  
else if (studentAnswer == 3) {  
    cout << feedback3 << endl;  
}  
else {  
    cout << feedback << endl;  
}  
  
if (studentAnswer == correctAnswer) {  
    points++;  
}  
  
cout << points << endl;
```
