CH 4: Making Decision

Topics Covered:

1. Relational Operators and Relational Expressions
2. Logical operators and Logical Expressions
3. if statement
4. if-else statement
5. nested if statement

Relational Operators and relational expressions

Numeric data is compared in C++ by using relational operators; Note that relational operators are binary operators. Recall: other binary operators: +, -, *, /, %

Neither operands will be modified due to a relational operator.

Relational Operators	Meaning
>	Greater than
<	less than
>=	Greater than or equal to
<=	less than or equal to
==	equal
!=	not equal

NOTE: == (Equal to Operator) and = (assignment Operator) are two different operators

An expression that uses relational operators are called relational expression

e.g.,

1. 3 > 4
2. 5 > 4
3. a >= 6
4. 7 < b
5. 1 == c
6. a <= b

Find the value of a relationship given below: assume a = 5, b = 15, c = 25, d = 6, e = 7Relational expression	Value of answer	output
answer = 3 > 4; cout << answer;	false	0
answer = (5 * a) > 4 cout << answer;	true	1
answer = a >= 6 cout << answer;	false	0
answer = 7 < (b +4) cout << answer;	true	
answer = 1 == c cout << answer;	false	
answer = (a -6) <= b cout << answer;	true	
answer = 5 != a cout << answer;	false	

Conclusion: A relational expression can be EITHER true or false; true represents 1 and false represents 0

The result of a relational expression can be assigned to a bool variable or to an integer variable. answer in the above examples, could be an int or bool variable.

Logical operators and Logical Expressions

Logical operators connect two or more relational expressions into one or ……

Operator	meaning	Effect	
&&	AND	Both expressions must be true for the overall expression to be true	Amy and Bonny will come to see you
||	OR	at least expressions must be true for the overall expression to be true	Amy or Bonny will come to see you.
!	NOT	reverse the truth of an expression	NOT (AMY) will come to see you.

a	b	a > 5 && b <=10
5	10	
5	9	
5	11	
4	10	
4	9	
4	11	
6	10	
6	9	
6	11	

a	b	a > 5 || b <=10
5	10	
5	9	
5	11	
4	10	
4	9	
4	11	
6	10	
6	9	
6	11	

ICA write a program to find the letter grade

if-else if statement

format:

```
if (expression)
   statement or block
else if (expression2)
   statement or block
else if (expression3)
   statement or block
    .
    :

else
   statement or block
```

e.g.,

```
double score;
cout << "Enter the score";
cin >> score;

if (score >= 90)
{
   cout << "A" << endl;
}
else if (score >= 80)
{
   cout << "B" << endl;
}
else if (score >= 70)
{
   cout << "C" << endl;
}
else
{
   cout << "F" << endl;
}
```

if statement

format:

```
if (expression)
   statement or block
```

e.g.,

```
double score;
cout << "Enter the score";
cin >> score;

if (score >= 90)
{
      cout << "Pass" << endl;
}
```

if-else statement

format:

```
if (expression)
   statement or block
else
   statement or block
```

e.g.,

```
double score;
cout << "Enter the score";
cin >> score;

if (score >= 90)
{
      cout << "Pass" << endl;
}
else
{
      cout << "Fail" << endl;

}
```

nested if statement: if statement is inside anther if statement:

```
double score;
cout << "Enter the score";
cin >> score;

if (score >= 90)
{
      cout << "A" << endl;
 }
else
{
      if (score >= 80)
      {
          cout << "B" << endl;
       }
       else
       {
           if (score >= 70)
           {
                cout << "C" << endl;
            }
            else
            {
                cout << "F" << endl;
            }
       }
}
```

Switch cases

Format of a Switch statement.

```
switch (option)
{
    case EXPRESSION1:
       //statements
    case EXPRESSION2:
       //statements
    case EXPRESSION3:
        // statements
    case EXPRESSION4:
       // statements
    default:
      //statements
}
```

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
       int dd;
       string day;

       cout << "Enter the month as a number";
       cin >> dd;


       switch (dd)
       {
         case 1:
              day = "Sunday";
              break;
         case 2:
              day = "Monday";
              break;

         case 3:
              day = "Tuesday";
              break;
         case 4:
              day = "Wednesday";
              break;
         case 5:
              day = "Thursday";
              break;
         case 6:
              day = "Friday" ;
              break;
```

```
          case 7:
               day = "Saturday";
               break;
          default:
               day = "Invalid";
       }

       cout << "day is: " << day;

       return 0;
}
```

- *option*:
    - should be a variable
    - Of any of the integer types, including char.
    - char, short, int, long, and long long,

C++ code (partial)	**validity**
int score; switch (score)	

switch(5)
{
  case 1:
  case 2:
  case 5:
}	

`int option,`
`    num3;`
`switch (option + num3)`	

bool options; cin >> options; switch (options)	

- EXPRESSIONS (e.g., EXPRESSION1, EXPRESSION2, …)
    - an expression whose value is of any of the integer data types.
    - Should be known at compile time.
      E.g., int num; cin >> num; would this num a valid expression?

C++ code (partial)	validity
const int ADD = 1; case ADD:	
case 1:	
case 1 + 0:	
case 1: case 1+ 0:	
`int num1;` case num1:	

C++ code (partial)	validity
const int ADD = 1; case ADD * 0:	

int x;
const int BLUE = 4;
switch (x)
{
case 4:
case BLUE:
}	

- *break* and default statement are optional.

When a case is matched, execution begins at the first statement following that label and continues until one of the following termination conditions is true (there are 4, but we need to learn only 2):

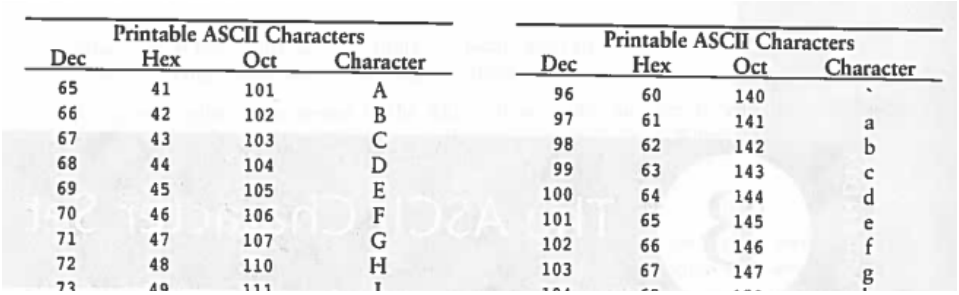
1) The end of the switch block is reached
2) A break statement occurs

If no case is matched, it executes the default statement, if available.

Comparing characters and strings

Comparing characters

Recall ASCII table:



Printable ASCII Characters			
Dec	Hex	Oct	Character
65	41	101	A
66	42	102	B
67	43	103	C
68	44	104	D
69	45	105	E
70	46	106	F
71	47	107	G
72	48	110	H

Printable ASCII Characters			
Dec	Hex	Oct	Character
96	60	140	`
97	61	141	a
98	62	142	b
99	63	143	c
100	64	144	d
101	65	145	e
102	66	146	f
103	67	147	g

```
char leterGrade1 = 'A',
      letterGrade2 = 'a';

if (leterGrade1 > leterGrade2)
    cout << "A > a";
else if (leterGrade1 < leterGrade2)
    cout << "A < a";
else
    cout << " A and a are equal";
```

output:

A < a

Comparing Strings

```
string name1 = "Mark",
       name2 = "Mary";

if (name1 > name2)
    cout << "Mark is bigger";
else if (name1 < name2)
    cout << "Mary is bigger";
else
   cout << "Mary and Mark are equal";
```

output:

Mary is bigger

Skipped: 4.13, 4:15.