CS 31 Week 3
Discussion 2E

Srinath

Announcements

- All **OH** and **Discussions** will be **in-person** going further!!
- Project 3 warmup is up! Due **11:00 PM, Wednesday, October 19**

Outline

- Switch statement
- Loops (while, do-while, for)
- Character
- Loop a String
- Worksheet 3

Switch Statement

Switch : revise If.. else..

Look at an if … else.. ladder.

```
if (Boolean_Expression_1)
    Statement_1
else if (Boolean_Expression_2)
    Statement_2
        .
        .
        .
else if (Boolean_Expression_n)
    Statement_n
else
    Statement_For_All_Other_Possibilities
```

When is Statement_1 executed?

Switch : revise If.. else..

Look at an if … else.. ladder.

```
if (Boolean_Expression_1)
    Statement_1
else if (Boolean_Expression_2)
    Statement_2
        .
        .
        .
else if (Boolean_Expression_n)
    Statement_n
else
    Statement_For_All_Other_Possibilities
```

When is Statement_1 executed?

- Boolean_Expression_1 is True

When is Statement_2 executed?

Switch : revise If.. else..

Look at an if … else.. ladder.

```
if (Boolean_Expression_1)
    Statement_1
else if (Boolean_Expression_2)
    Statement_2
        .
        .
        .
else if (Boolean_Expression_n)
    Statement_n
else
    Statement_For_All_Other_Possibilities
```

When is Statement_1 executed?

- Boolean_Expression_1 is True

When is Statement_2 executed?

- Boolean_Expression_1 is False and
- Boolean_Expression_2 is True

When is Statement_n executed?

Switch : revise If.. else..

Look at an if … else.. ladder.

```
if (Boolean_Expression_1)
    Statement_1
else if (Boolean_Expression_2)
    Statement_2
        .
        .
        .
else if (Boolean_Expression_n)
    Statement_n
else
    Statement_For_All_Other_Possibilities
```

When is Statement_1 executed?

- Boolean_Expression_1 is True

When is Statement_2 executed?

- Boolean_Expression_1 is False and
- Boolean_Expression_2 is True

When is Statement_n executed?

- Boolean_Expression_1, Boolean_Expression_2,…, Boolean_Expression_n-1 are False and
- Boolean_Expression_n is True

When is Statement_For_All_Other_Possibilities executed?

Switch : revise If.. else..

Look at an if … else.. ladder.

```
if (Boolean_Expression_1)
    Statement_1
else if (Boolean_Expression_2)
    Statement_2
        .
        .
        .
else if (Boolean_Expression_n)
    Statement_n
else
    Statement_For_All_Other_Possibilities
```

When is Statement_1 executed?
- Boolean_Expression_1 is True

When is Statement_2 executed?
- Boolean_Expression_1 is False and
- Boolean_Expression_2 is True

When is Statement_n executed?
- Boolean_Expression_1, Boolean_Expression_2,…, Boolean_Expression_n-1 are False and
- Boolean_Expression_n is True

When is Statement_For_All_Other_Possibilities executed?
- Boolean_Expression_1, …, Boolean_Expression_n are False

Switch :

```
switch(expression) {
   case constant-expression  :
      statement(s);
      break; //optional
   case constant-expression  :
      statement(s);
      break; //optional

   // you can have any number of case statements.
   default : //Optional
      statement(s);
}
```

default is executed if expression do not match any of the cases.

Switch :

```
switch(expression) {
   case constant-expression  :
      statement(s);
      break; //optional
   case constant-expression  :
      statement(s);
      break; //optional

   // you can have any number of case statements.
   default : //Optional
      statement(s);
}
```

default is executed if expression do not match any of the cases.

```
int vehicleClass;
double toll;
cout << "Enter vehicle class: ";
cin >> vehicleClass;
switch (vehicleClass)
{
   case 1:
      cout << "Passenger car.\n";
      toll = 0.50;
      break;
   case 2:
      cout << "Bus.\n";
      toll = 1.50;
      break;
   case 3:
      cout << "Truck.\n";
      toll = 2.00;
      break;
   default:
      cout << “Unknown Vehicle.\n”;
   }
```

Switch :

```
switch(expression) {
   case constant-expression  :
      statement(s);
      break; //optional
   case constant-expression  :
      statement(s);
      break; //optional

   // you can have any number of case statements.
   default : //Optional
      statement(s);
}
```

default is executed if expression do not match any of the cases.

```
int vehicleClass;
double toll;
cout << "Enter vehicle class: ";
cin >> vehicleClass;
switch (vehicleClass)
{
   case 1:
      cout << "Passenger car.\n";
      toll = 0.50;
      break;
   case 2:
      cout << "Bus.\n";
      toll = 1.50;
      break;
   case 3:
      cout << "Truck.\n";
      toll = 2.00;
      break;
   default:
      cout << “Unknown Vehicle.\n”;
   }
```

What if we remove break statement in case 2 and the input vehicleClass is 2?

Switch :

```
switch(expression) {
   case constant-expression  :
      statement(s);
      break; //optional
   case constant-expression  :
      statement(s);
      break; //optional

   // you can have any number of case statements.
   default : //Optional
      statement(s);
}
```

default is executed if expression do not match any of the cases.

```
int vehicleClass;
double toll;
cout << "Enter vehicle class: ";
cin >> vehicleClass;
switch (vehicleClass)
{
   case 1:
      cout << "Passenger car.\n";
      toll = 0.50;
      break;
   case 2:
      cout << "Bus.\n";
      toll = 1.50;
      break;
   case 3:
      cout << "Truck.\n";
      toll = 2.00;
      break;
   default:
      cout << “Unknown Vehicle.\n”;
   }
```

What if we remove break statement in case 2 and the input vehicleClass is 2?

- Bus
  Truck

- **toll** will be 2.00

Switch : if → switch

```
if (x == 10)
{
   cout << "Foo" << endl;
}
else if (x == 20)
{
   x = 30;
}
else if (x == 30)
{
   x = 100;
}
else
{
   cout << "Bar" << endl;
}
```

```
switch( ...? ) {

}
```

Switch : if → switch

```
if (x == 10)
{
   cout << "Foo" << endl;
}
else if (x == 20)
{
   x = 30;
}
else if (x == 30)
{
   x = 100;
}
else
{
   cout << "Bar" << endl;
}
```

```
switch( x ) {
        case ...? :
}
```

Switch : if → switch

```
if (x == 10)
{
   cout << "Foo" << endl;
}
else if (x == 20)
{
   x = 30;
}
else if (x == 30)
{
   x = 100;
}
else
{
   cout << "Bar" << endl;
}
```

```
switch( x ) {
        case 10 :
                cout << “Foo” << endl;
}
```

Switch : if → switch

```
if (x == 10)
{
   cout << "Foo" << endl;
}
else if (x == 20)
{
   x = 30;
}
else if (x == 30)
{
   x = 100;
}
else
{
   cout << "Bar" << endl;
}
```

```
switch( x ) {
        case 10 :
                cout << “Foo” << endl;
        case ...? :
}
```

Switch : if → switch

```
if (x == 10)
{
   cout << "Foo" << endl;
}
else if (x == 20)
{
   x = 30;
}
else if (x == 30)
{
   x = 100;
}
else
{
   cout << "Bar" << endl;
}
```

```
switch( x ) {
        case 10 :
                cout << “Foo” << endl;
        case 20 :
                x = 30;
}
```

Switch : if → switch

```
if (x == 10)
{
   cout << "Foo" << endl;
}
else if (x == 20)
{
   x = 30;
}
else if (x == 30)
{
   x = 100;
}
else
{
   cout << "Bar" << endl;
}
```

```
switch( x ) {
        case 10 :
                cout << “Foo” << endl;
        case 20 :
                x = 30;
        case 30 :
                x = 100;
}
```

Switch : if → switch

```
if (x == 10)
{
   cout << "Foo" << endl;
}
else if (x == 20)
{
   x = 30;
}
else if (x == 30)
{
   x = 100;
}
else
{
   cout << "Bar" << endl;
}
```

```
switch( x ) {
        case 10 :
                cout << “Foo” << endl;
        case 20 :
                x = 30;
        case 30 :
                x = 100;
        default :
                cout << “Bar” << endl;
}
```

Switch : if → switch

```
if (x == 10)
{
   cout << "Foo" << endl;
}
else if (x == 20)
{
   x = 30;
}
else if (x == 30)
{
   x = 100;
}
else
{
   cout << "Bar" << endl;
}
```

```
switch( x ) {
        case 10 :
                cout << “Foo” << endl;
                break;
        case 20 :
                x = 30;
                break;
        case 30 :
                x = 100;
                break;
        default :
                cout << “Bar” << endl;
}
```

Switch : if → switch

```
if (c == 'A' || c == 'B')
{
  cout << "Two cases in one!" <<
  endl;
}
else if (c == 'C')
{
  cout << c << endl;
}
```

```
switch( c ) {
        ...?
}
```

Switch : if → switch

```
if (c == 'A' || c == 'B')
{
  cout << "Two cases in one!" <<
  endl;
}
else if (c == 'C')
{
  cout << c << endl;
}
```

```
switch( c ) {
        case 'A' :
                cout << "Two cases in one!" << endl;
                break;
        case 'B' :
                cout << "Two cases in one!" << endl;
                break;
        case 'C' :
                cout << c << endl;
}
```

Switch : if → switch

```
if (c == 'A' || c == 'B')
{
  cout << "Two cases in one!" <<
  endl;
}
else if (c == 'C')
{
  cout << c << endl;
}
```

```
switch( c ) {
        case 'A' :
                cout << "Two cases in one!" << endl;
                break;
        case 'B' :
                cout << "Two cases in one!" << endl;
                break;
        case 'C' :
                cout << c << endl;
}
```

Looks like 'A' and 'B' are doing the same thing.
Can we combine them?

Switch : if → switch

```
if (c == 'A' || c == 'B')
{
  cout << "Two cases in one!" <<
  endl;
}
else if (c == 'C')
{
  cout << c << endl;
}
```

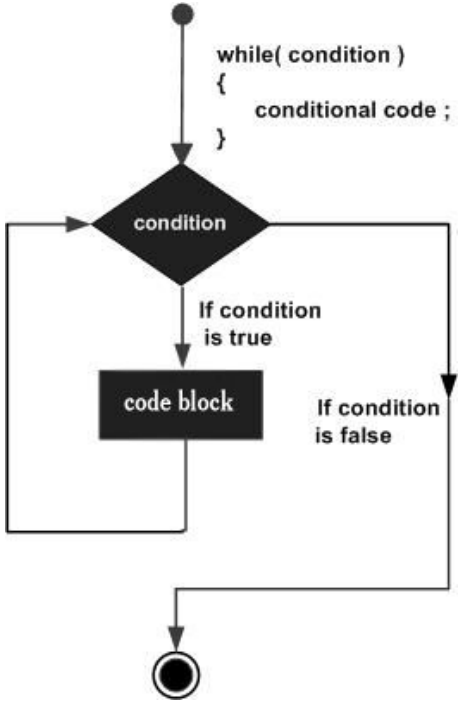
```
switch( c ) {
        case 'A' :
        case 'B' :
                cout << "Two cases in one!" << endl;
                break;
        case 'C' :
                cout << c << endl;
}
```

Loops

Loops : while

```
while(condition) {
   statement(s);
}
```

The boolean expression(**condition**) is checked **before** the loop body is executed.



Loops : while

```
while(condition) {
    statement(s);
}
```

The boolean expression(**condition**) is checked **before** the loop body is executed.

```
1  #include <iostream>
2  using namespace std;

3  int main( )

4  {
5      int countDown;

6      cout << "How many greetings do you want? ";
7      cin >> countDown;

8      while (countDown > 0)
9      {
10         cout << "Hello ";
11         countDown = countDown - 1;
12     }

13     cout << endl;
14     cout << "That's all!\n";

15     return 0;
16 }
```

Output when input is 3?

Loops : while

```
while(condition) {
    statement(s);
}
```

The boolean expression(**condition**) is checked **before** the loop body is executed.

```
1  #include <iostream>
2  using namespace std;

3  int main( )

4  {
5      int countDown;

6      cout << "How many greetings do you want? ";
7      cin >> countDown;

8      while (countDown > 0)
9      {
10         cout << "Hello ";
11         countDown = countDown - 1;
12     }

13     cout << endl;
14     cout << "That's all!\n";

15     return 0;
16 }
```

Output when input is 3?

- Hello Hello Hello
  That's all!

Output when input is 0?

Loops : while

```
while(condition) {
    statement(s);
}
```

The boolean expression(**condition**) is checked **before** the loop body is executed.

```
1  #include <iostream>
2  using namespace std;

3  int main( )

4  {
5      int countDown;

6      cout << "How many greetings do you want? ";
7      cin >> countDown;

8      while (countDown > 0)
9      {
10         cout << "Hello ";
11         countDown = countDown - 1;
12     }

13     cout << endl;
14     cout << "That's all!\n";

15     return 0;
16 }
```

Output when input is 3?

- Hello Hello Hello
  That's all!

Output when input is 0?

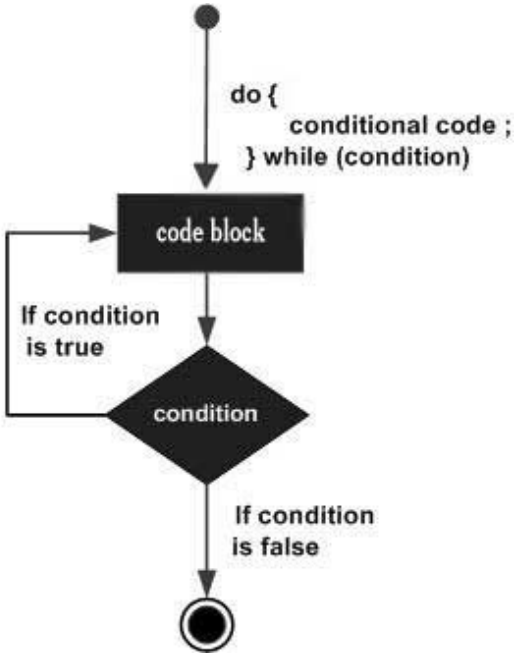
-
  That's all!

Loops : do-while

```
do {
   statement(s);
}
while( condition );
```

The boolean expression(**condition**) is checked **after** the loop body is executed.

Executes loop body at least once.



Loops : do-while

```
do {
   statement(s);
}
while( condition );
```

The boolean expression(**condition**) is checked **after** the loop body is executed.

Executes loop body at least once.

```
1  #include <iostream>
2  using namespace std;

3  int main( )
4  {
5      int countDown;

6      cout << "How many greetings do you want? ";
7      cin >> countDown;

8      do
9      {
10         cout << "Hello ";
11         countDown = countDown - 1;
12     } while (countDown > 0);

13     cout << endl;
14     cout << "That's all!\n";

15     return 0;
16 }
```

Output when input is 3?

Loops : do-while

```
do {
   statement(s);
}
while( condition );
```

The boolean expression(**condition**) is checked **after** the loop body is executed.

Executes loop body at least once.

```
1  #include <iostream>
2  using namespace std;

3  int main( )
4  {
5      int countDown;

6      cout << "How many greetings do you want? ";
7      cin >> countDown;

8      do
9      {
10         cout << "Hello ";
11         countDown = countDown - 1;
12     } while (countDown > 0);

13     cout << endl;
14     cout << "That's all!\n";

15     return 0;
16 }
```

Output when input is 3?

- Hello Hello Hello That's all!

Output when input is 0?

Loops : do-while

```
do {
   statement(s);
}
while( condition );
```

The boolean expression(**condition**) is checked **after** the loop body is executed.

Executes loop body at least once.

```
1  #include <iostream>
2  using namespace std;

3  int main( )
4  {
5      int countDown;

6      cout << "How many greetings do you want? ";
7      cin >> countDown;

8      do
9      {
10         cout << "Hello ";
11         countDown = countDown - 1;
12     } while (countDown > 0);

13     cout << endl;
14     cout << "That's all!\n";

15     return 0;
16 }
```

Output when input is 3?

- Hello Hello Hello
  That's all!

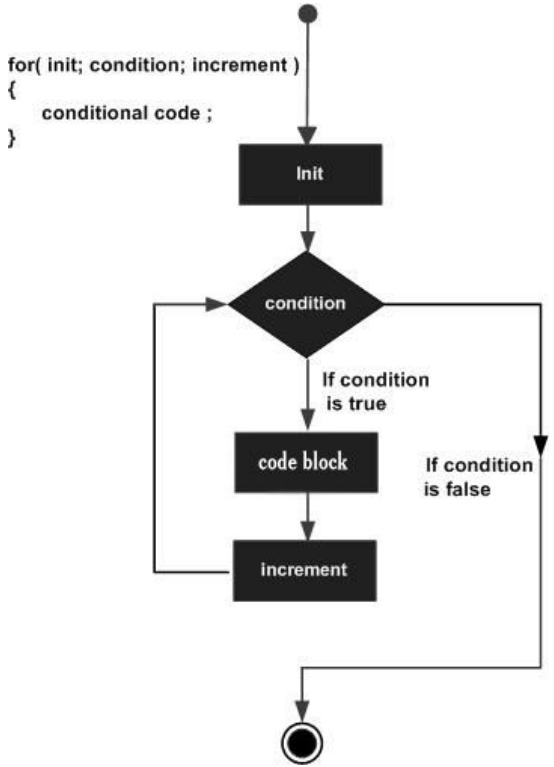
Output when input is 0?

- Hello
  That's all!

Loops : for

```
for ( init; condition; increment ) {
   statement(s);
}
```

```
int sum = 0;
for (int n = 1; n <= 10;  n++)
     sum = sum + n;
cout << "Sum is: " << sum << endl;
```



Note: In C++, a variable may be declared in the heading of a for statement so that the variable is both declared and initialized at the start of the for statement. But the ANSI/ISO C++ standard requires that a C++ compiler treats it as a local variable to the body of the loop.

Loops : for → while

```
for (num = 100; num >= 0;
num--)
{    cout << num << " bottles of
     beer on the wall.\n";
}
```

??

Loops : for → while

```
for (num = 100; num >= 0;
num--)
{    cout << num << " bottles of
     beer on the wall.\n";
}
```

```
num = 100;
while(num>=0)
{
   cout << num << " bottles of
   beer  on the wall.\n";
   num--;
}
```

Loops : for → do-while

```
for (num = 100; num >= 0;
num--)
{    cout << num << " bottles of
     beer on the wall.\n";
}
```

??

Loops : for → do-while

```
for (num = 100; num >= 0;
num--)
{    cout << num << " bottles of
     beer on the wall.\n";
}
```

```
num =
100;  do
{
   cout << num << " bottles of
   beer  on the wall.\n";
   num--;
}while(num>=0);
```

Loops : for → do-while

```
for (num = 100; num >= 0;
num--)
{    cout << num << " bottles of
     beer on the wall.\n";
}
```

```
num =
100;  do
{
   cout << num << " bottles of
   beer  on the wall.\n";
   num--;
}while(num>=0);
```

Be careful when you do this, sometimes may not get same results in all cases. Say num is -1 instead?

Character

Character : Definition

Character constant

‘A’, ‘B’, ‘a’, ‘@’, ‘^’ etc. are characters.

We use single quotes to represent a character constant.

Character variable

Character is also a type like int, double

We can declare

```
char a;
char b = ‘h’;
char multiply = ‘*’;
```

Character : Definition

Character constant

‘A’, ‘B’, ‘a’, ‘@’, ‘^’ etc. are characters.
We use single quotes to represent a character constant.

Character variable

Character is also a type like int, double
We can declare

```
char a;
char b = ‘h’;
char multiply = ‘*’;
```

Learn more about ASCII value
at the ASCII Chart

ASCII Value

- In C++, every character is associated with an Integer called it’s ASCII value. This is for internal representation.
- If we assign ‘h’ to a char variable, 104 is stored in the variable.

Characters	ASCII Values
A	65
Z	90
a	97
z	122
5	53

Character : ASCII

Get ASCII Value of a Character

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

int main() {
    char ch = 'h';

    // Printing the corresponding ASCII of a character
    // Notice the use of int() to get an integer
    cout << "ASCII value = " << int(ch) << endl;

    return 0;
}
```

Output:

```
Character = 104
```

Print Character Using ASCII Value

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

int main() {

    // assigning an integer value to char
    char ch = 104;

    // printing the variable
    cout << "Character = " << ch << endl;

    return 0;
}
```

Output:

```
Character = h
```

Character : Escape sequences

Even ‘ and “ are also characters, In C++ some characters have special meaning like ‘, “, \ etc. and we cannot use them directly in our program.

```
// This code shows an error
char character = ''';
```

Character : Escape sequences

Even ‘ and “ are also characters, In C++ some characters have special meaning like ‘, “, \ etc. and we cannot use them directly in our program.

```
// This code shows an error
char character = ''';
```

C++ provides us special codes known as escape sequences.

```
// does not show error
char character = ' \' ';
```

Character : Escape sequences

Escape Sequences	Characters
\b	Backspace
\f	Form feed
\n	Newline
\r	Return
\t	Horizontal tab
\v	Vertical tab
\\	Backslash
\'	Single quotation mark
\"	Double quotation mark
\?	Question mark
\0	Null Character

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

int main() {
    char character1 = 'A';

     // using escape sequence for horizontal tab
    char character2 = '\t';

    char character3 = '5';

     // using escape sequence for new line
    char character4 = '\n';

    char character5 = 'a';

    // printing the variables
    cout << character1;       // A
    cout << character2;       // horizontal tab
    cout << character3;    // 5
    cout << character4;    // new line
    cout << character5;   // a

    return 0;
}
```

Output?

Character : Escape sequences

Escape Sequences	Characters
\b	Backspace
\f	Form feed
\n	Newline
\r	Return
\t	Horizontal tab
\v	Vertical tab
\\	Backslash
\'	Single quotation mark
\"	Double quotation mark
\?	Question mark
\0	Null Character

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

int main() {
    char character1 = 'A';

     // using escape sequence for horizontal tab
    char character2 = '\t';

    char character3 = '5';

     // using escape sequence for new line
    char character4 = '\n';

    char character5 = 'a';

    // printing the variables
    cout << character1;       // A
    cout << character2;       // horizontal tab
    cout << character3;    // 5
    cout << character4;    // new line
    cout << character5;   // a

    return 0;
}
```

Output?

A 5
a

Looping a String

Looping a String :

A string is a sequence of characters.
Eg: “Hello” is a sequence of ‘H’, ‘e’, ‘l’, ‘l’, ‘o’

Each character in a string has an index increasing from left to right, starting from 0.
For string “Hello”, we have at index
0 - ‘H’
1 - ‘e’
2 - ‘l’
3 - ‘l’
4 - ‘o’

It is a 5 character string and index ranges from 0 to 4.

Looping a String :

A string is a sequence of characters.
Eg: “Hello” is a sequence of ‘H’, ‘e’, ‘l’, ‘l’, ‘o’

Each character in a string has an index increasing from left to right, starting from 0.
For string “Hello”, we have at index
0 - ‘H’
1 - ‘e’
2 - ‘l’
3 - ‘l’
4 - ‘o’

It is a 5 character string and index ranges from 0 to 4.

string name = “srinath”;

Accessing strings

.at(i) - to get character at **i** th index
name.at(2) gives us **i**

.size() - to get length of string
name.size() gives us **7**

Looping a String :

As a string is a sequence of characters, we can loop through the string and access each of its characters in an orderly fashion.

```
string name = “srinath”;
for(int i=0 ; i != name.size(); i++) {
        cout << name.at(i) <<endl;
}
```

Prints the string, character by character

Looping a String :

As a string is a sequence of characters, we can loop through the string and access each of its characters in an orderly fashion.

```
string name = “srinath”;
for(int i=0 ; i != name.size(); i++) {
        cout << name.at(i) <<endl;
}
```

Prints the string, character by character

Counting number of A’s (both lower and upper case)

Looping a String :

As a string is a sequence of characters, we can loop through the string and access each of its characters in an orderly fashion.

```
string name = “srinath”;
for(int i=0 ; i != name.size(); i++) {
        cout << name.at(i) <<endl;
}
```

Prints the string, character by character

Counting number of A’s (both lower and upper case)

```
string name = “srinath”;
int numOfAs = 0;
for(int i=0 ; i != name.size(); i++) {
        If (name.at(i)==‘a’ || name.at(i)==‘A’ ) {
                numOfAs = numOfAs+1;
        }
}

cout << numOfAs << endl;
```

Questions??