Mittuniversitetet
MID SWEDEN UNIVERSITY

Lecture #2
Elementary Programming

Chapter: 2, 19.6

Objectives

Mittuniversitetet
MID SWEDEN UNIVERSITY

- To use identifiers to name elements such as variables.
- To write C++ programs that perform simple computations.
- To read input from the keyboard.
- To name constants using the **const** keyword.
- To declare variables using numeric data types.
- To write integer literals, floating-point literals.
- To use augmented assignment operators (**+=**, **-=**, ***=**, **/=**, **%=**).
- To distinguish between post-increment and pre-increment and between post-decrement and pre-decrement.
- To convert numbers to a different type using casting.
- To obtain the current system time using **time(0)**.

Identifiers

Mittuniversitetet
MID SWEDEN UNIVERSITY

- An identifier is a sequence of characters that consists of letters, digits, and underscores (_).
- An identifier must start with a letter or an underscore. It cannot start with a digit.
- An identifier cannot be a reserved word
- An identifier can be of any length, but your C++ compiler may impose some restriction. Use identifiers of 31 characters or fewer to ensure portability.

Declaring Variables

Mittuniversitetet
MID SWEDEN UNIVERSITY

```
int x;          // Declare x to be an integer variable;

double radius; // Declare radius to be a double variable;

char a;         // Declare a to be a character variable;
```

Variables

Mittuniversitetet
MID SWEDEN UNIVERSITY

```
// Compute the first area
radius = 1.0;
area = radius * radius * 3.14159;

// Compute the second area
radius = 2.0;
area = radius * radius * 3.14159;
```

Assignment Statements

Mittuniversitetet
MID SWEDEN UNIVERSITY

```
x = 1;          // Assign 1 to x;
radius = 1.0;   // Assign 1.0 to radius;
a = 'A';        // Assign 'A' to a;
```

Declaring and Initializing

```
int x = 1;
double d = 1.4;
```

Named Constants

Mittuniversitetet
MID SWEDEN UNIVERSITY

Syntax:

```
const datatype CONSTANTNAME = VALUE;
```

Example:

```
const double PI = 3.14159;
const int SIZE = 3;
```

A complete example

Mittuniversitetet
MID SWEDEN UNIVERSITY

Computing the Area of a Circle

This program computes the area of the circle.

ComputeArea

Reading Input from the Keyboard

Mittuniversitetet
MID SWEDEN UNIVERSITY

You can use the **cin object** to read input from the keyboard.

ComputeAreaWithConsoleInput

Reading Multiple Input in One Statement

Mittuniversitetet
MID SWEDEN UNIVERSITY

ComputeAverage

sizeof Operator

Mittuniversitetet
MID SWEDEN UNIVERSITY

You can use the **sizeof operator** to find the size of a type. For example, the following statement displays the size of int, long, and double on your machine.

```
cout << sizeof(int) << " " << sizeof(long) << " " << sizeof(double);
```

Example: Displaying Time

Mittuniversitetet
MID SWEDEN UNIVERSITY

Write a program that obtains hours and minutes from seconds.

DisplayTime

Augmented Assignment Operators

Operator	*Example*	*Equivalent*
+=	i += 8	i = i + 8
-=	f -= 8.0	f = f - 8.0
*=	i *= 8	i = i * 8
/=	i /= 8	i = i / 8
%=	i %= 8	i = i % 8

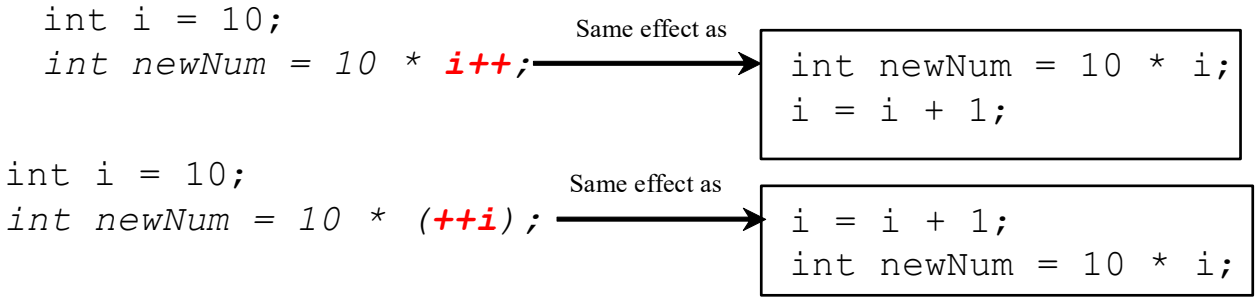
Increment and Decrement Operators

Mittuniversitetet
MID SWEDEN UNIVERSITY

Operator	Name	Description
++var	preincrement and evaluates	The expression (++var) increments var by 1 to the *new* value in var *after* the increment.
var++	postincrement *original* value	The expression (var++) evaluates to the in var and increments var by 1.
--var	predecrement and evaluates	The expression (--var) decrements var by 1 to the *new* value in var *after* the decrement.
var--	postdecrement *original* value	The expression (var--) evaluates to the in var and decrements var by 1.

Increment and Decrement Operators, cont.

Mittuniversitetet
MID SWEDEN UNIVERSITY



Using increment and decrement operators makes expressions short, but it also makes them complex and difficult to read. Avoid using these operators in expressions that modify multiple variables, or the same variable for multiple times such as this: int k = ++i + i.

Type Casting

Mittuniversitetet
MID SWEDEN UNIVERSITY

Implicit casting:

double d = 3; (type widening)

Explicit casting:

int i = static_cast<int>(3.0); (type narrowing)

int i = (int)3.9; (Fraction part is truncated)

Casting does not change the variable being cast. For example, d is not changed after casting in the following code:

```
double d = 4.5;
int i = static_cast<int>(d);  // d is not changed
```

Example: Keeping Two Digits After Decimal Points

Mittuniversitetet
MID SWEDEN UNIVERSITY

Write a program that displays the sales tax with two digits after the decimal point.

SalesTax

Using predefined functions

Mittuniversitetet
MID SWEDEN UNIVERSITY

Solve the quadratic equation $\mathbf{ax^2+bx+c = 0}$ for input values a, b and c.

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Predefineds

Thank you