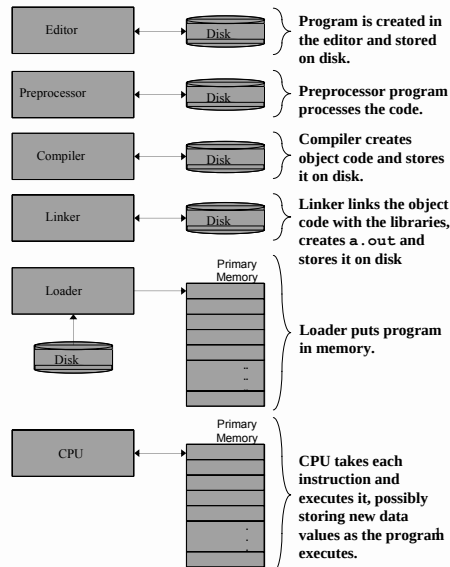


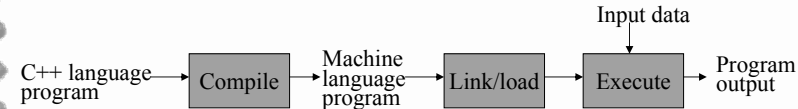
Basics of a Typical C++ Environment

Phases of C++ Programs:

1. Edit
2. Preprocess
3. Compile
4. Link
5. Load
6. Execute



Executing a Computer Program



Compiler

- Converts **source** program to **object** program

Linker

- Converts **object** program to **executable** program

2

Basics of a Typical C++ Environment

C++ systems

- Program-development environment
- Language
- C++ Standard Library

3

A Basic C++ Program Structure

```
/* *****  
 * Name: Ali Saif  
 * Class: COMP 2002 Section 20  
 * ID: 123456  
 * Assignment #: 1  
 * Date: 10 February, 2007  
 * Description: This program will add two integers and display  
 *             their sum on the screen.  
 * *****/  
  
#include <iostream>  
using namespace std;  
int main ()  
{  
    // write your code here  
    return 0;  
}
```

4

#include Preprocessor Command

☞ Copies source code into the program from the specified file.

☞ #include <iostream>

- Contains **class** information for input and output.
- Processed by preprocessor before compiling
- Begin with **#**

5

Namespace

- **std::**: specifies using name that belongs to “namespace” **std**
- **std::**: removed through use of **using** statements

```
std::cout;  
std::cin
```

```
using namespace std;
```

```
....
```

```
.....
```

```
cout <<  
cin >>
```

6

Basics of a Typical C++ Environment

☞ Input/output

- **cin**
 - Standard input stream
 - Normally keyboard
- **cout**
 - Standard output stream
 - Normally computer screen

7

Simple I/O - **cin**

☞ **cin**

- is an istream object
- streams input from standard input
- uses the >> (input operator)

General Form:

```
cin >> identifier >> identifier;
```

Note: Data entered from the keyboard must be compatible with the data type of the variable.

8

Simple Output - cout

cout

- is an ostream object
- streams output to standard output
- uses the << (output) operator

General Form:

cout << *expression* << *expression*;

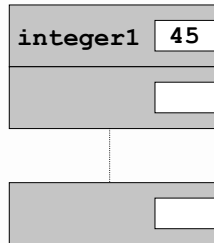
Note: An expression is any C++ expression (string constant, identifier, formula or function call)

9

Memory Concepts

Variable names

- Correspond to actual locations in computer's memory
- Every variable has name, type, size and value
- When new value placed into variable, overwrites previous value
- Reading variables from memory nondestructive



10

Variable Declarations

Declarations define memory locations, including type of data to be stored, identifier, and possibly an initial value.

General Form:

data_type identifier_list;

11

Variable Declarations

Declarations define memory locations, including type of data to be stored, identifier, and possibly an initial value.

General Form:

data_type identifier_list;

Examples:

double length(20.75), width(11.5), volume;
int numberOfFeetInYard(3);

12

C++ Data Types

Integer numbers: **int**, **short**, **long**

Floating point numbers: **float**, **double**

Characters: **char**, **string**

Boolean (true/false): **bool**

13

Example Data-Type Limits

Data type	Maximum value
short	32,767
int	2,147,483,647
float	6 digits of precision Maximum exponent = 38 Maximum value = 3.402823e +38
double	15 digits of precision Maximum exponent = 308 Maximum value = 1.797693e +308

14

Naming entities in C++

Identifiers are used to name entities in C++.

Rules for construction of identifiers

- Start with a letter or underscore _
- Consist of letters digits and underscore
- Can not be a reserved word.
- Only first 31 characters used to distinguish it from other identifiers.
- Case sensitive

15

C++ keywords

– Cannot be used as identifiers or variable names

C++ Keywords

Keywords common to the C and C++ programming languages

auto	break	case	char	const
continue	default	do	double	else
enum	extern	float	for	goto
if	int	long	register	return
short	signed	sizeof	static	struct
switch	typedef	union	unsigned	void
volatile	while			
C++ only keywords				
asm	bool	catch	class	const_cast
delete	dynamic_cast	explicit	false	friend
inline	mutable	namespace	new	operator
private	protected	public	reinterpret_cast	
static_cast	template	this	throw	true
try	typeid	typename	using	virtual
wchar_t				

16

Variable Naming Examples

exam1	Valid
budget\$	Invalid
2tests	Invalid
first_name	Valid
short	Invalid

Note: exam1 and Exam1 are not the same variable¹⁷

Variable Declaration Examples

```
int exam1;
```

```
int exam1, exam2;
```

```
int exam1=0;    OR    int exam1(0);
```

```
char ch;
```

```
ch = 'a';
```

18

Arithmetic Operators

Addition	+
Subtraction	-
Multiplication	*
Division	/
Modulus	%

- Modulus returns remainder of division between two *integers*
- Example
5%2 returns a value of 1

19

Example 2

- Write a C++ program which will prompt the user to enter two integers and display their sum.

20

Integer Division

Division between two integers results in an integer.

The result is truncated, not rounded

Example:

$5/3$ is equal to 1

$3/6$ is equal to 0

21

Priority of Operators

1. Parentheses Inner most first
2. Unary operators Right to left
(+ -)
3. Binary operators Left to right
(* / %)
4. Binary operators Left to right
(+ -)

22

Precedence of Arithmetic Operators

Expression	Value
$3 - 4 * 5$	-17
$3 - (4 * 5)$	-17
$(3 - 4) * 5$	-5
$3 * 4 - 5$	7
$3 * (4 - 5)$	-3
$17 - 10 - 3$	4
$17 - (10 - 3)$	10
$(17 - 10) - 3$	4
$-42 + 50 \% 17$	-26
$17 / 3$	5

23

Precedence of Arithmetic and Assignment Operators

Precedence	Operator	Associativity
1	Parentheses: ()	Innermost first
2	Unary operators + - ++ -- (type)	Right to left
3	Binary operators * / %	Left to right
4	Binary operators + -	Left to right
5	Assignment operators = += -= *= /= %=	Right to left

24

Precedence of Arithmetic Operators

Algebra: $m = \frac{a + b + c + d + e}{5}$

C++: ~~`(a + b + c + d + e) / 5;`~~

Algebra: $y = mx + b$

C++: `y = m * x + b;`

25

Example 3

Write a computer program to calculate the volume of a box based on its length, width and height. The program should allow the user to enter the length, width and height of the box and it should calculate the corresponding volume and display the result.

26

Example 3 Solution

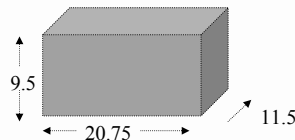
1. Write a program to compute the volume of box.

2. Input: length, width, height

Output: volume

3. Hand Example:

volume = $20.75 * 11.5 * 9.5$



4. Algorithm: input length, width and height
compute volume
output volume

5. Implement algorithm in C++ and test.

27

Declare 3 variable of type integer to store length, width, height

Declare 1 variable of type integer to store volume

Ask the user to enter length

Read length and store it in variable length

Ask the user to enter width

Read width and store it in variable width

Ask the user to enter height

Read height and store it in variable height

Calculate volume and store the result in variable volume

volume = length * width * height

Print volume on the screen

28

```

*****
* Name: Ali Saif
* Class: COMP 2002 Section 20
* ID: 123456
* Assignment #: 1
* Date: 10 February, 2007
* Description: This program will calculate the volume of a box
               and display the result on the screen
*
*****/
#include <iostream>
using namespace std;

int main()
{
    // Declare and initialize objects
    double length;
    double width;
    double height;
    double volume;
    // get the user input
    cout << "Enter the length of the box: ";
    cin >> length;
    cout << "Enter the width of the box: ";
    cin >> width;
    cout << "Enter the height of the box: ";
    cin >> height;

    // Calculate volume.
    volume = length * width * height;
    // Print the volume.
    cout << "The volume is " << volume << endl;

    // Exit program.
    return 0;
}

```

29

Exercise

Write a C++ program which will prompt the user to enter a temperature in celsius and convert it to fahrenheit. The temperature will entered using the keyboard. Once the conversion calculation is completed the new temperature will be displayed on the screen in fahrenheit.

30

Note:

$$C = (5/9) * (F-32)$$

$$F = (9/5) * C + 32$$

31

Symbolic Constants

- ☞ Used to name values which do not change during the execution of the program.
- ☞ Are always initialized at declaration.
- ☞ Used wherever an expression is allowed.

General Form:

const *data_type identifier* = *value*;

32

Symbolic Constant

```
const double PI = 3.14;  
const double PI= acos(-1.0);  
area= PI * radius * radius;  
  
const double gravity= 9.80665
```

33

Exercise

Write a C++ program to calculate the area of a circle. The user will be prompted to enter the diameter of the circle using the keyboard. The program will then calculate the area and display the result on the screen.

34

Increment and Decrement Operators

Increment Operator ++

- post increment `x++;` $\rightarrow x=x+1$
- pre increment `++x;`

Decrement Operator --

- post decrement `x- -;` $\rightarrow x=x-1;$
- pre decrement `--x;`

35

Increment and Decrement Operators

`x++` or `++x` $\rightarrow x=x+1$

`x--` or `--x` $\rightarrow x=x-1$

What is the difference between `x++` and `++x` ?

`w= ++x -y;`

`x=x+1;`

`w=x-y;`

`W=x++ -y;`

`w=x-y;`

`x = x+1;`

For examples assume `k=5` prior to executing the statement.

- `m= ++k;` both `m` and `k` become 6
- `n = k- -;` `n` becomes 5 and `k` becomes 4

36

Abbreviated Assignment Operator

operator example equivalent statement

+=	x+=2;	x=x+2;
-=	x-=2;	x=x-2;
=	x=y;	x=x*y;
/=	x/=y;	x=x/y;
%=	x%=y;	x=x%y;

For Example:

x=x+3; ➔ x+=3;
x = x * 2; ➔ x*=2;

37

Precedence of Arithmetic and Assignment Operators

Precedence	Operator	Associativity
1	Parentheses: ()	Innermost first
2	Unary operators + - ++ -- (type)	Right to left
3	Binary operators * / %	Left to right
4	Binary operators + -	Left to right
5	Assignment operators = += -= *= /= %=	Right to left

38

Scientific Notation

0.0004 = 4.0 x 10⁻⁴ = 4.0e-4
-254.3 = -2.543 x 10² = -2.443e2
-23256 = -2.3256 x 10⁴ = -2.3256e4

In C++, there are two type of notations:

- Fixed notation e.g. (0.0004)
- Scientific notation e.g. (4.0e-4)

39

Manipulators and methods

- endl – places newline character in stream *and flushes the buffer.*

Flag	Meaning
ios::showpoint	display the decimal point
ios::fixed	fixed decimal notation
ios::scientific	scientific notation
ios::right	right justification
ios::left	left justification

- Manipulators in <iomanip>

–setprecision(n)
–setw(n)

40

Stream Functions

❏ `cout.setf` //set format
❏ `cout.setf(ios::fixed);`
❏ `cout.setf(ios::showpoint);`
❏ `cout.precision(n);` //n digits after the decimal point

`cout << setprecision(n)` //n digits after the decimal point
`cout << setw(n)` //specifies the width of variables to n positions

❏ See example in Pages 47-49

41

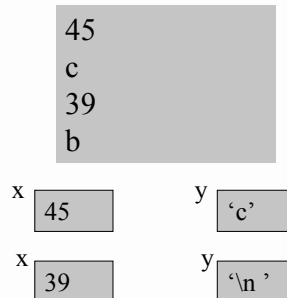
Functions in <cmath>

<code>abs(x)</code>	computes absolute value of x
<code>sqrt(x)</code>	computes square root of x, where $x \geq 0$
<code>pow(x,y)</code>	computes x^y
<code>ceil(x)</code>	nearest integer larger than x
<code>floor(x)</code>	nearest integer smaller than x
<code>exp(x)</code>	computes e^x
<code>log(x)</code>	computes $\ln x$, where $x > 0$
<code>log10(x)</code>	computes $\log_{10} x$, where $x > 0$
<code>sin(x)</code>	sine of x, where x is in radians
<code>cos(x)</code>	cosine of x, where x is in radians
<code>tan(x)</code>	tangent of x, where x is in radians

42

Characters and input

❏ `>>` discards leading whitespace
❏ `get()` method used to input whitespace characters
❏ Example:
`int x;`
`char y;`
`cin >> x >> y;`
`cin >> x;`
`cin.get(y);`



43

Exercise

Write a C++ program which asks for the user's name and print the following message

Hello *user name*

Welcome to C++

Note: *user name*, is what the user will enter using the keyboard.

44

Exercise

```
int x = 2, y(0);  
cout << "x= " << x << endl;  
x++;  
cout << "x= " << x << endl;  
++x;  
cout << "x= " << x << endl;  
y = ++x;  
cout << "y= " << y << "    x= " << x << endl;  
y = x++;  
cout << "y= " << y << "    x= " << x << endl;
```