Lecture 1 Notes

Statements

A program contains a sequence of statements. A statement either produces an identifier or a value. An identifier is a name given to some entity inside the program. A value is an instance of data that can undergo computation.

Data types

A value is an instance of data. All values have a data type. A value's data type determines how the value is stored, interpreted and manipulated. Here are five data types supported by C++:

Data type	Description	Size	Range
`bool`	Boolean	1 byte	`true`, `false`
`char`	Character	1 byte	-128 to 127
`int`	Integer	4 bytes	-2147483648 to 2147483647
`double`	Real number	8 bytes	-1.7E308 to 1.7E308
`string`	Character sequence	–	–

Variables

A variable is a container that can store a value. A variable can be referenced by it's identifier. A C++ statement that creates a variable is known as a variable declaration.

Listing 1: Examples of variable declarations

```
int x;
double amount;
bool cointoss;
```

A variable declaration is performed by specifying the data type and the identifier of the variable. You may optionally provide an initialization value to your new variable, for example:

Listing 2: Creating variables with initialization values

```
double balance = 3070.12;
bool verified = true;
string message = "hello world"s;
```

If you do not provide an initialization value, the variable assumes a default value depending on it's data type:

Data type	Default value
`bool`	`false`
`char`	`'\0'`
`int`	`0`
`double`	`0.0`
`string`	`""s`

Expressions

An expression produces a value. Expressions are either literal values or a composition of operators and operands.

Literal value expressions

Literal values represent constant instances of data with implicit data types:

Data type	Examples of literal values
`bool`	`false, true`
`char`	`'A', '3', '?', ' ', '\n', '\t'`
`int`	`12, 0, -23, 500, +1234`
`double`	`9.8, -0.678, 23.0E+3, -0.1E-5`
`string`	`"welcome"s, "access denied."s, "Time\tPosition\n"s`

Most expressions are composed of operators and operands. An operand is a sub-expression used by an operator, and an operator performs an operation on it's operands.

Assignment expressions

You have already been introduced to the simple assignment operator:

Operator Name	Operator Symbol	Usage
Simple assignment	`=`	`a = b`

The simple assignment operator assigns the value of it's right operand to it's left operand. This operator returns it's left operand, therefore you can assign a value to multiple variables in one expression:

Listing 3: Examples of assignment expressions

```
x = 4;
foo = bar;
a = b = c = 5;
```

The last expression is equivalent to: `c = 5; b = c; a = b;`.

Arithmetic expressions

Arithmetic operators are typically used on the `int` and `double` data types.

Operator Name	Operator Symbol	Usage
Addition	`+`	`a + b`
Subtraction	`-`	`a - b`
Multiplication	`*`	`a * b`
Division	`/`	`a / b`
Modulo	`%`	`a % b`

These operators are self-explanatory. The modulo operator is similar to the division operator except that it returns the remainder instead of the quotient.

I/O expressions

I/O stands for Input/Output. An I/O operator transfers data on a stream. A stream is a data type that represents a communication channel.

Operator Name	Operator Symbol	Usage
Insertion	<<	a << b
Extraction	>>	a >> b

The insertion operator writes the value from it's right operand to the stream on it's left operand. The extraction operator reads a value from the stream on it's left operand, and stores that value into the variable on it's right operand.

Both of these operators return their left-operand, therefore you can perform multiple insertions or multiple extractions on the same stream in one expression.

Standard Streams

Every C++ program has access to two pre-declared stream variables. These are known as the program's standard streams.

Variable	Description	Default target
cin	Character in	Keyboard
cout	Character out	Terminal window

You can use the extraction operator on `cin` to read user input into a variable. You can use the insertion operator on `cout` to write messages into the terminal window.

Hello World

This is C++ source code for a hello world program. A hello world program displays a simple message and exits.

Listing 4: hello.cpp

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

int main()
{
    cout << "hello world\n"s;
}
```

To run this program, you must save this source code into a file named hello.cpp. Then you must compile hello.cpp into a program and run it:

Listing 5: Bash commands to compile and run hello.cpp

```bash
g++-5 -std=c++14 hello.cpp
./a.out
```

The first bash command compiles hello.cpp to generate an executable file named a.out. The second command executes a.out.

– Mark Swoope