UCLA Samueli
Computer Science

CS31 Week 2 Discussion

Fall 2021, Section 1C

Mingyu Derek Ma mdma@ucla.edu

Thanks Muhao Chen, Rosa Garza for their shared content

https://derek.ma/cs31 for slides and other discussion materials

Reminder

- Make sure you can run program on Linux
  - Linux workshop by LA
- Project 2: Thursday Oct 14th
- zyBook programming assignment 2: Saturday Oct 9th

Read in a string

- `cin` only take the first part of input separated by newline or space
- `getline()` will take everything in the input until the next newline
- `getline()` right after using cin
  - Only takes left text in previous cin input and the new line
- You could use `cin.ignore`(num of characters to discard, what character to ignore) to discard cached characters
  - `cin.ignore(10000, '\n')`
  - `cin.ignore(10000, ' ')`

Program example 1, 2, 3

Constants

- A constant cannot be changed or be reassigned, it contains variable that useful for the entire program.

- const int NUM_DISCUSSIONS = 8;

Operators

- % modulo
  - Modulo operator computes remainder
  - `25 % 20 = 5`
- Arithmetic operators
  - * / % has higher precedence
  - + - has lower precedence
- Examples
  - a = 11 * 3; // a is 33
  - b = 11(2+3); //compile error
  - b = 11 * (2+3); //

Data types in arithmetic operation

- If at least 1 value is a double, then the output value will be a double
- If double being stored to int value, it is truncated (rounded down)

```
int firstVal;
double secVal;
firstVal = 10.0/4; // firstVal is 2
secondVal = 10.0/4; // secondVal is 2.5
```

- If final value is double (ex: 12.0), it does not print out decimal with 0s. If you want to print those 0s, you can set the precision

```
cout.setf(ios::fixed);
cout.setf(ios::showpoint);
cout.precision(2); // number of significant digits after decimal
```

Program example 4

Compound assignment

- `+=, -=, *=, /=, %=`
- Examples: left and right are equivalent

```
count += 2;
total -= discount;
bonus *= 2;
time /= rushFactor;
amount *= c1 + c2;
```

```
count = count + 2;
total = total - discount;
bonus = bonus * 2;
time = time/rushFactor;
amount = amount * (c1 + c2);
```

Incremental operators

- `++, --`
- Examples: left and right are equivalent

```
int i = 1;

i++;
i--;
```

```
int i = 1;

i = i + 1;
i = i - 1;
```

Incremental operators

- `++, --`
- Incremental operators in assignment statement
- `++i` first increases the value of i, and then return the increased value
- `i++` returns the current value first, and then do increment

```
int i = 1;
j = ++i;
// i is 2, j is 2
// Equal to
// i += 1;
// j = i;
```

```
int i = 1;
j = i++;
// i is 2, j is 1
// Equal to
// j = i;
// i += 1;
```

Program example 5

Comparison operators

- Mathematical comparison
  - ==
  - !=
  - <
  - <=
  - >
  - >=
- Logical comparison
  - &&
    - and
  - ||
    - or
  - !
    - negation
- Printing out booleans
  - With Xcode/Visual C++, booleans will be printed as 1/0
  - You can use `cout.setf(ios::boolalpha)` to print boolean value as True/False

Condition Execution guarantees

- Then there are multiple logical operators, && has higher precedence than ||
  - `if ((major == “CS” || major == ”MATH”) && gpa >= 3.2)`
  - `if (major == “CS” || major == ”MATH” && gpa >= 3.2)`
- For A && B, evaluate A first, if A is true, then evaluate B
  - If A is false, then B is not even evaluated
  - Order does matters
  - `if (b != 0 && d != 0 && a/b + c/d < 10)`
  - `if (a/b + c/d < 10 && b != 0 && d != 0) // logic error`
- For A || B, evaluate A first, if A is false then evaluate B, result is A or B
  - if A us true, result is true, B is not evaluated at all

If else

- Use if-else statements when the program needs to perform different behaviors under different conditions
- Format 1: if and else
  - Define a block by `{}` to use multiple statement
  - Variables declared inside a block is only useful within that block

```
if(boolean expression)
      statements;
else
      statements;
```

Program example 6

```
if(boolean expression){
      statements1;
      statements2;
} else {
      statements;
}
```

If else

- Format 2: Only if
  - Use {} to protect the code block with only if statement
  - Or add an empty else
    - else ;

Program example 7

If else

- “else if” simplification
  - It’s not a special statement, just a simplified version of if else statements

```
if(boolean expression)
      statement;
else if(boolean expression)
      statement;
else if(boolean expression)
      statement;
else
      statement;
```

```
if(boolean expression)
      statement;
else
      if(boolean expression)
            statement;
      else
            if(boolean expression)
                  statement;
            else
                  statement;
```

Program example 8

Other expression as condition

- If the condition is integer 0, then it’s treated as False if you need a T/F value
- If the condition is non-zero integer, then it’s treated as True if you need a T/F value
- If the condition is an assignment statement, the expression will put value to the variable and then produce new value of this variable. So in T/F value condition, it’s equal to the new value of the variable
  - Like `n = 23;`

Program example 9

Thank You