Complexity

Bryce Boe

2013/10/21

CS24, Fall 2013

Outline

- Compiler Review
- Project 1 Questions
- Complexity (Big-O)
- C++?

COMPILER REVIEW

What does the following produce?

clang foobar.c

a.out
(executable)

What does the following produce?

clang -c foobar.c

foobar.o
(object file)

What does the following produce?

clang foobar.o

a.out
(executable)

What does the following produce?

clang foobar.c blah.c

a.out
(executable)

What does the following produce?

clang -g foobar.c

a.out
(executable, debuggable)

What does the following produce?

clang foobar.c -o foo

foo
(executable)

PROJECT 1 INFO

Resubmissions until 2PM Wednesday

- Keep working on project 1
  - Getting through it completely will help tremendously on the other projects
- I have an office hour tomorrow
- Don't forget about lab 4 (due Tuesday night)
- We'll go over a solution on Wednesday

COMPLEXITY

Space v. Time

- Space complexity is how much storage relative to the input does an algorithm require
- Time complexity is how many computations relative to the input does an algorithm require
- In computing we often trade space for time or time for space to get the desired performance

Think about it

- How much (extra) **space** is required to compute the sum of a list of numbers?

constant
(perhaps 1 int)

Think about it

- How much **time** is required to compute the sum of a list of numbers?

linear
(must sum up each)

Think about it

- How much (extra) **space** is required to sort a list of numbers?
- How much **time?**

varies depending on implementation

Think about it

- How much **time** is required to see if an item is contained in a sorted list?

logarithmic

Big-O

- In complexity analysis we typically care about the worst-case
- We also consider the input as it grows to infinity
- We refer to this worst-case as **Big-O**, or **O(?)**

Computing Big-O

- If something has an actual running time function of:

- $T(\mathbf{n}) = 2\mathbf{n}^3 + 100\mathbf{n}^2 + 1024\mathbf{n} + 10\text{trillion}$
- $O(T(\mathbf{n})) = O(\mathbf{n}^3)$

- Consider only the fasted growing term, and remove any factors.

Common Ordered Complexities

- O(1) – constant
- O(log(n)) – logarithmic
- O(n) – linear
- O(nlog(n)) – linearithmic
- $O(n^2)$ – quadratic
- $O(2^n)$ – exponential
- O(n!) – factorial

O(?)

```
int a[1024];  // assume allocated and not static

if (a[0] == a[1])
        return a[2];
else
        return a[0];
```

O(1)

O(?)

```
int a[4];  // assume allocated and not static
int n = sizeof(a) / sizeof(int);
int sum = 0;
for (int i = 0; i < n; ++i)
        sum += a[i];
return sum;
```

O(n)

O(?)

```
int a[4];  // assume allocated

return a[0] + a[1] + a[2] + a[3];
```

O(1)

O(?)

```
int a[1024];  // assume allocated
int n = sizeof(a) / sizeof(int);
int dups = 0;
for (int i = 0; i < n; ++i)
        for (int j = 0; j < n; ++j)
                if (a[i] == a[j])
                        ++dups;
```

$O(n^2)$

O(?)

```
int a[1024];  // assume allocated
int n = sizeof(a) / sizeof(int);
int dups = 0;
for (int i = 0; i < n; ++i)
        for (int j = i; j < n; ++j)
                if (a[i] == a[j])
                        ++dups;
```

$O(n^2)$

C++ INTRODUCTION

Why C++?

- Problems with C
  - Has a single global namespace
  - Cannot use the same name for functions with different types (e.g., min(int, int) and min(double, double)) – called *overloading*
  - Difficult to minimize source-code repetition for similar functions with different types

Some Differences

- #include <stdio.h> → #include <iostream>
  - Or if you want fprintf, etc use #include <cstdio>
- printf("Hello\n"); → cout << "Hello\n";
- Rather than defining a **struct** which only contains data, define a **class** which contains data and methods on the data
- **throw** exceptions rather than use return values to represent error cases

Classes

- Provide encapsulation
  - Combining a number of items, such as variables and functions, into a single package, such as an object of some class (or instance of the class)

Scope Resolution Operator

- ClassName::method_name
- Used to identify the scope, class in this case, that the method belongs to as there may be more than 1 instance of method_name
- Scope resolution isn't necessary if you are also a member of that class