Clemson Hack Pack

Clemson ACM

August 30, 2014

10 Commandments of ACM Contests

Paraphrased from Dr. Dean

1. Thou shalt sort first and ask questions later
2. Thou shalt know the STL and use it well
3. Thou shalt know thy algorithms by heart
4. Thou shalt brute force $\leq$ 10 million items
5. Thou shalt when in doubt solve with DP
6. Thou shalt never count by 1
7. Thou shalt reinitialize thy data structures
8. Thou shalt test often and submit early
9. Thou shalt never trust a sample input
10. Thou shalt print according to the output spec

Remember what the Dr. said
"Algorithms are Cool!"

Contents

Chapter 1

Basic Data Structures

1.1 Set

Sets are Associative, Ordered, set, Unique Keyed, and Allocator Aware[1]. Set means that the key is also the value. Use it when it is only important if something has been seen before.

```
#include <set>
std::set<type_one> myset;

//iterator
std::set<type_one>::iterator it;

//insert O(N log(N)) or O(N) am for sorted inputs
myset.insert(type_one (value));

//remove O(log N) or O(1) am post-find
myset.erase(it);
myset.erase(key);

//find O(log N)
myset.find(key)
```

1.2 Map

Maps are Associative, Ordered, Mapped, Unique Keyed, and Allocator aware[1]. Mapped means that each key corresponds to a specific value. Use it to record relationships in data.

```
#include <map>
std::map<key_type,value_type> mymap;

//iterator
std::map<key_type, value_type>::iterator it;

//insert O(N log(N)) or O(N) am for sorted inputs
mymap.insert( std::pair<key_type, value_type>(key, value);
mymap[key] = value;

//remove O(log N) or O(1) am post-find
mymap.erase(it);
mymap.erase(key);
```

```
//find O(log N)
mymap.find(key)
```

1.3 Heaps

Chapter 2

Algorithms

2.1 Dijkstra

Chapter 3

C IO Functions

Occasionally it is far easier to use the C IO functions to meet an output spec. It is also possible to set the precision and width options via numbers after the present,but before the specifier. The general form of a specifier is:

```
%[flags][width][.precision][length]specifier
```

Table 3.1: Format Specifier Codes [1]

Format Code	Output	Example
d	Signed Int	314
u	Unsigned Int	314
o	Unsigned Octal	472
x	Unsigned hex	13a
X	UNSIGNED HEX	13A
f	floating point	3.140000
e	Scientific notation	3.140000e+00
c	character	A
s	string	ACM
p	pointer address	0x40060c
l	Used with other specifiers to indicate a long	314
%%	Prints a literal %	%

Table 3.2: Modifier Flags [1]

Format Code	Output	Example
-	Left-justify	314
+	Force-sign character	+314
#	Show prefix	0x13a
	Show decimal point	314.
0	Left pad field with 0	0314

3.1 Examples

```
#include <stdio.h>
int main (){
    //To stdout
    double f = 3.14;
```

```
    int    i = 314;
    char*  s = "ACM";
    printf("%5f,%5i,%5s\n", f, i, s);

    //Same thing to a file
    FILE * outputfile;
    outputfile = fopen("outputfile.txt","w");
    fprintf(outputfile,"%5f,%5i,%5s\n", f, i, s);
    fclose(outputfile);
}
```

Chapter 4

Appendix

4.1 Some Basic VIMRC Settings

```
set mouse=a
imap jj $<ESC>$
set ai
set nu
set scs
set bs=2
set ts=4
sy on
colo=slate
set bg=dark
```

4.2 Makefile

```
CC=g++
all :
    $(CC) *.c -o $*.o
```

Bibliography

[1] The C++ Resources Network. cplusplus.cpm, 2013.