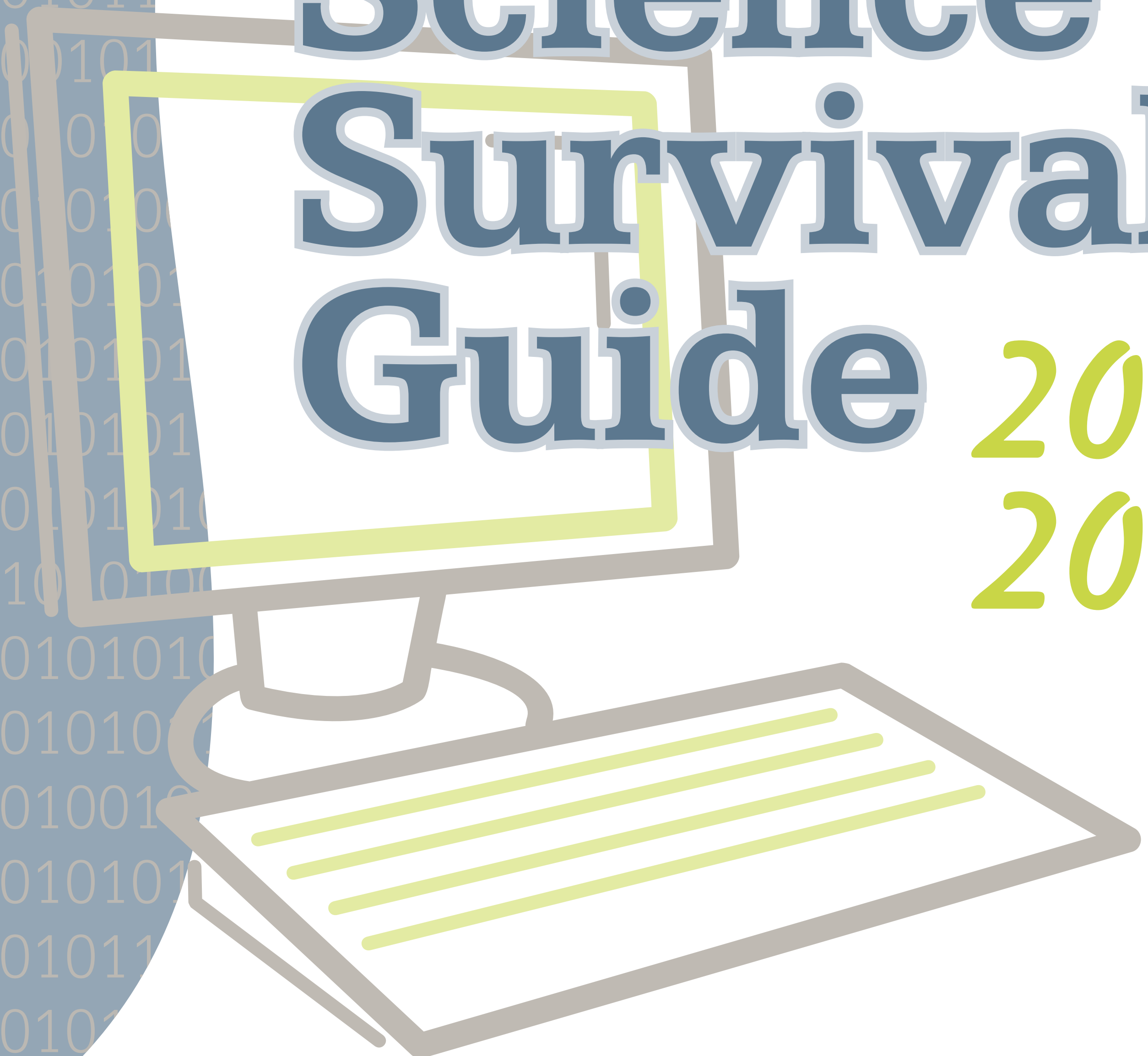
Carleton Computer Science Society

~presents~

Computer Science Survival Guide 2006-2007



Contents

Tip Sorting out the abbreviations

Ever find yourself confused by the many abbreviations thrown at you? Here’s a guide to the basics.

CCSS	Carleton Computer Science Society
CUUC	Carleton University Undergraduate Calendar
SCS	School of Computer Science
BCS	Bachelor of Computer Science
TA	Teaching Assistant
Raven	Carleton's mascot, but more often SCS's online assignment submission system

Here are some codes for buildings where most computer science students have class sooner or later.

HP	Herzberg Physics building, home of SCS, CCSS, and student computer labs
ME	Mackenzie Building, engineering building
TB	Tory Building, home of the Egg
SA	Southam Hall home of the Alumni Theatre
AT	Azrieli Theatre, home of several large teaching theatres

Top Five

The tools you'll need

5 English writing skills

Having good skills in both spoken and written English is essential, even in the world of computer science. You need to be able to communicate with your teaching assistants, professors, and group members. Using formal English when emailing professors often gets you more respect and, ultimately, help with your problems. You will also be writing large reports, such as for your honours project. To up your writing skills, take some first year English classes. Look into CCDP 2000 (Communication Skills for Computer Science Students), an excellent course for all computer science students to take.

4 A good backpack

Don't underestimate how much you'll end up carrying around campus. After you fill a bag with a laptop, lunch, notebooks, and text books, your back will probably start to protest. Renting a locker is a good way to lighten the load for at least part of the day, but a good backpack can help you survive the rest. Choose something a bit on the smaller side to help force yourself to carry less, and go for a model with chest and waist straps if you want to save yourself from back pain even more.

3 CCSS Web Site

The CCSS is always adding more valuable resources to their site (found at http://ccss.carleton.ca/), so be sure to check it for updates regularly. This guide includes a detailed description of what the CCSS site will offer.

2 Raven

This assignment submission system is used for most computer science classes. While it can be a bit tricky to use, it sure beats submitting printed source code and floppy disks. Be sure you know how to use Raven early in your career at Carleton, and any time you are using it from a particular computer for the first time, be ready to submit a day early to leave time for resolving any issues. Look for tutorials on the web page after you sign in (the main page is at http://www.scs.carleton.ca/raven/).

1 Popular Integrated Development Environments (IDEs)

Even if you already have a favourite IDE, you should learn more than one of the most popular choices, such as Microsoft Visual Studio and Eclipse. In particular, you should be sure you can debug your programs quickly and easily. This will not only help make courses with strict OS and language requirements easier, but it will make you that much more employable. You can still use your favourite when the choice is yours. Look on the MSDNAA web site to find out how to download free student copies of Microsoft products (login with your Connect account, http://msdnaa.carleton.ca). Then check out Eclipse, the popular Java IDE, available free (http://www.eclipse.org/downloads/).

Tips

Surviving the HP labs

While some of us may try to work on our own computers at home as much as possible, there is often no way around working in the school's labs during group projects. Spending hours on end working on assignments or studying for exams definitely takes its toll. Here are some tips for making the experience a bit more enjoyable.

• **Listen to music.** If you don't have an MP3 player, just buy some cheap headphones and plug them into the lab computers. There are countless online radio stations you can tune into (including The Bear from Ottawa, as just one example). You can also try personal services that learn what you like (check out pandora.com for one).

• **Don't go hungry.** Don't let fears of the dreaded Frosh Fifteen stop you from snacking (you can always munch on healthy food if you are worried). Keeping your brain fed is sure to maintain both the quality of your work and your mood. It is probably better to bring your own snacks to save money and avoid eating chocolate bars every day, but for those times you don't have anything, visit the CCSS lounge or Carleton's many vending machines and cafeterias for meals and snacks, healthy and otherwise.

• **Take stretch breaks.** It's very easy to become so involved with what you're doing that time slips away. But you need to make sure you get up and move around at least twice every hour. Your muscles will thank you for the chance to change position. Your eyes also need the rest. Drink lots of water to force yourself to go to the bathroom frequently if you tend to forget to take breaks.

• **Take longer exercise breaks.** If you're in for a long study session, plan to take an hour or two to hit the gym, go for a long walk, or otherwise exercise. This will give your mind a chance to think about something else, making the time you spend in the lab upon return much more effective. As an added bonus, you don't have to feel so guilty about snacking!

• **Work when others are already in the lab.** CCSS will have volunteers in their office during peak hours. When there are enough students around, the society may organize ordering pizza.

• **If you happen to be working late**, keep in mind that stores are not open, bus schedules are reduced, and the doors will lock behind you!

Top Five Things to do before you graduate

5 Attend a Discovery Lecture

Each year, a guest speaker is invited to do a free public lecture sponsored by the College of Natural Sciences and the School of Journalism and Communication. The subjects of these lectures are scientific in nature, in the past covering topics from the weather to living in space and on Mars. Learn something new and discover just how relevant science is to our everyday lives! Keep your eye on the Faculty of Science's web site event listing for this year's exciting lecture (news can be found online here http://www.carleton.ca/science/).

4 Learn a new tool or programming language

Enhance your abilities with any programming language other than the mainstream choices. For instance, you could learn Smalltalk or Turing. Or, pick a development tool you've never used before (a profiler, for instance) and become an expert.

3 Obtain your lab access card

Not all labs in HP are designated to only computer science students, and not all labs require access cards. However, the labs that do need cards tend to be less busy and therefore may have equipment that is in better condition. Be sure to visit University Safety in 203 Robertson Hall (close to the Coke machine) to obtain your card. Find out more about the various labs, including their hours, online (http://www.scs.carleton.ca/nethelp/labs.php).

2 Get to know your profs

Not only will this make going to class that much easier, but you'll probably find out very quickly that you are interested in the research of at least a few of them. It will also make finding an honours project supervisor trivial and give you a great contact if you are eventually interested in grad school. The CCSS holds several events each year where you can chat with professors in a casual setting.

1 Attend a geeky conference

Not only are these conferences fun to attend with fellow CS students, but they can show involvement with the technology community on your resume, which many employers look for. Check out the Canadian Undergraduate Technology Conference in Toronto every January (www.cutc.ca), events held by the Ottawa Canada Linux User's Group (http://www.oclug.on.ca), the Linux Symposium held in Ottawa every year in July (http://www.linuxsymposium.org), the Desktop Developer's Conference also held in Ottawa in July (http://www.desktopcon.org), or the Ottawa Venture and Technology Summit in Gatineau in October (http://www.ottawavts.com).The SCS and CCSS often sponsor students to attend events like these.

Feature

What's online at CCSS

You can come visit the CCSS lounge anytime in 4135 HP. In the meantime, check out the web site (http://ccss.carleton.ca) for many useful services. Remember that the site's usefulness is increased with the number of students that use and contribute to it.

CCSS news and events. Any time the CCSS puts on, participates in, or promotes an event, you'll find out about it here.

Lounge information. Learn about the services offered in the lounge, including snack prices, course notes for sale, and our informal tutorial services. You can also find out the times that volunteers do office hours to sell the snacks and notes.

Executive information. Find out how to contact the current CCSS executive and how to volunteer to help with events or do office hours in the lounge.

Anonymous feedback. You'll have a chance to give compliments or suggestions for improvement for any professor without having your identity revealed. The CCSS will pass this information on to the SCS in order to help offer a better education experience to students.

Exam questions database. When students leave an exam, they can jot down whatever questions they can remember and add them to the database. That way, students in the future will be able to have a few questions to practice while studying.

Course discussion forums. There is a forum for each course offered in the School of Computer Science. Here you can ask questions about course content, tests, or assignments for all students to answer. Also, professors and teaching assistants will be encouraged to frequent these forums and answer your questions.

General discussion forums. Anything goes in these forums. They are a good place to talk about anything from current events to obscure programming languages to why cheddar is the best cheese. There are also buy/sell forums, and forums to discuss co-op and high tech jobs.

Tips
How to spend less money

• **Don't buy books you don't need.** Computer science and math books are very expensive. It's a good idea to wait until the first day of class before purchasing your text books. This way, you can see whether the books are mandatory, or whether the material is available online. You may also be able to determine whether the professor's notes will be sufficient for you to learn the course content, but generally you should not rely on this.

• **Buy the books you do need used.** Carleton's book store often has used copies of the books you need, and you can also look for students trying to sell their books on the CCSS web site. Be sure to check student-run Haven Books, which may have some of the books you need at a lower price. It can be a bit more difficult to find computer science books there, though.

• **Follow student recommendations for books if you still aren't sure.** While every student learns differently, many suggest that these books are among to most worthwhile to purchase: Introduction to Algorithms (COMP 3804/4804), Theory of Computation (COMP 2805), C Programming Language, C++ Programming Language, and any book from the "In a Nutshell" series.

• **When buying a computer, consider carefully what you really need.** If you are really tight on cash, use the computers at school, which are guaranteed to have what you need to finish your homework. Otherwise, buy the most powerful computer you can afford, and make sure you can upgrade it later on. Avoid computers with preloaded bloat and avoid laptops when the budget is limited because they are much more expensive when they are small enough to be conveniently carried around school all day. Check out OEM Express for good deals on new hardware, and Computer Recyclers for decent used hardware. Finally, keep in mind that extra memory is useful for compiling but you don't really need fancy video cards and such for schoolwork.

• **Choose the right operating system or systems.** Consider choosing Linux so you will have access to multitudes of useful free software. Be sure to download your free student version of Microsoft Windows from MSDNAA (login here with your Connect ID: http://msdnaa.carleton.ca). Note that for most courses, you have a choice in what operating system you use, although you must speak with your TA to be sure. Windows is more easily accessible at school from the labs than Linux, but both are available.

• **Make use of the athletics facilities.** When you pay tuition, you pay for use of Athletics. Your membership includes use of the pool and weight room, for example. Working out right on campus saves time and money compared to going to another gym.

Top Five Succeeding academically

5 Choose your electives carefully.

In some streams, there are not many free electives available, while in others the number can seem daunting. In either case, you should choose classes for each elective very carefully. If you don't play on your strengths, you may find your elective grades bringing down your overall CGPA. For example, if your experimental science skills are weak, you may want to take PHYS 1901 (Planetary Astronomy) or BIOL 1902 (Natural History) for the science credit instead of traditional physics or chemistry. If you have a knack for languages, try taking French classes like FREN 1100 (First Year University French) or learn a whole new language like Japanese in classes like JAPA 1201 (Intensive Introductory Japanese). If logic is like second nature to you, try PHIL 2001 (Introduction to Symbolic Logic).

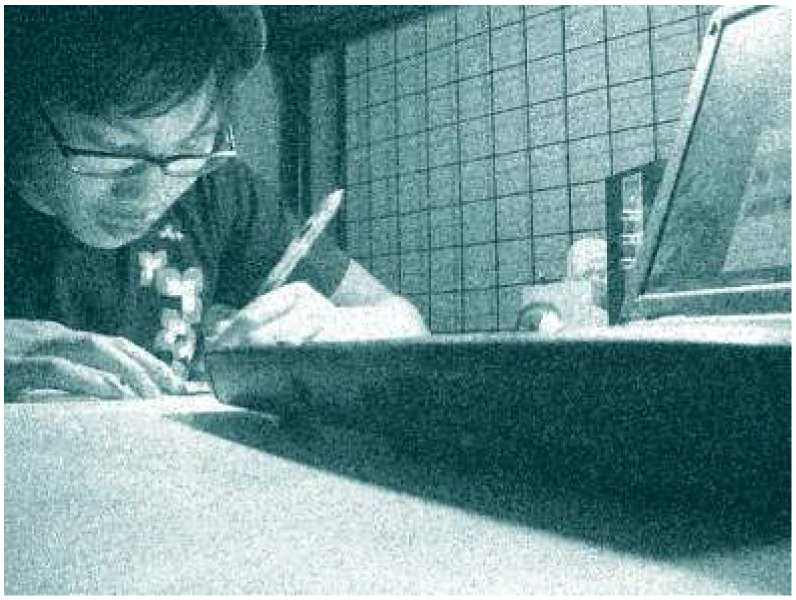
4 Communicate with professors and TA's promptly.

If you are unhappy with any grades you have received, you must speak with the professor (or teaching assistant if they did the grading) immediately. There is a good chance your mark can be changed, but only if you act quickly. Remember to present your case politely and provide solid reasoning on why there was a mistake or you deserve better. Don't become bitter or continue arguing if the professor disagrees with you.

3 Bring course notes to class.

Some professors make their course notes available online before class. This will NOT help you skip classes. Professors almost always give more information in class than can be found in the notes. Furthermore, reading through the material once will not help you remember it. Instead, read the notes before or after class, and use a printed copy to help you pay attention to the professor's explanations during class. You will often want to add your own notes and examples as well. If you don't want to use up your ink on printing the notes, bring a laptop to class or buy a set of printed notes from the CCSS.

(continued...)



Top Five

Succeeding academically

2 Tackle assignments logically.

It is generally a bad idea to start on the first question of an assignment, and move on to the next only when the current one is finished. Questions are not always ordered from easiest to hardest, and each question may only be worth a few marks, so spending a lot of time on any one could put your grade for the entire assignment in jeopardy. Instead, spend a bit of time looking at each question, jotting down a few ideas for each, and tackle them in order of how well you know the answer so far and how much each is worth. For programming assignments, learn early how to test effectively, and how to show that you have done so. Always remember to include a readme file to help the TA understand your work.

1 Understand that your hand will not be held!

It's your responsibility to attend class, read your text books, and finish your assignments. It only matters to you whether you do well or not.So take an interest in what you are doing, even if you are finding yourself bored in, say, a first year Java class when you've already been programming Java for years. If the immediate course material doesn't interest you, visit your professor and find out more about it. Or, take the opportunity to ace the class and take assignments to the next level. There's always something new to learn if you look for it, and taking an interest beyond the classroom is a key aspect to succeeding in the classroom.If, on the other hand, you find the material of a class too hard, be sure to ask lots of questions, and use the resources available to you. Visit teaching assistants and professors during their office hours, and they will be pleased to work through the tough material. Go to the Math Tutorial Center for problems with mathematics courses. Find a tutor on the CCSS web page for some extra one-on-one instruction. Do whatever it takes, but don't be lazy and let it go because you'll regret it later when the going gets even tougher.

What do my grade points mean?

(Letter grade, percentage, grade points)

A+	90-100%	12.0
A	85-89%	11.0
A-	80-84%	10.0
B+	77-79%	9.0
B	73-76%	8.0
B-	70-72%	7.0
C+	67-69%	6.0
C	63-66%	5.0
C-	60-62%	4.0
D+	57-59%	3.0
D	53-56%	2.0
D-	50-52%	1.0
F	0-49%	0.0

Find out more at http://www.carleton.ca/cu0607uc/regulations/acadregsuniv2.html

Tips

Increase your employability

When looking for high tech jobs both during school and after graduation, there are certain things that you can do to make yourself stand out to employers. These are just but a few tips; check out the CCSS web site's discussion forum for articles and advice on finding work in this industry.

- **Take relevant classes early.** If you know what kind of work you may want to do during the summer or for co-op, take classes with related content as early as you can. This way employers can easily see that you have learned the skills needed for the job, which helps when you don't have relevant past experience.

- **Always have a resume ready.** You never know when you might talk to a job contact. Always have an up to date resume on your computer and keep a printed copy in your book bag

- **Have your resume edited by several people.** Nothing scares away employers like a typo or bad grammar.

- **Practice your interview skills.** Succeeding in an interview is not as easy as it sounds. There is a lot to know about how to present yourself and what to say, not to mention how to prepare for technical questions. Look for help at the Co-op Office (http://www.carleton.ca/co-op/) or Career Services (http://www.carleton.ca/career/).

- **Get involved.** Employers want to see that you participate in the high tech community and have other interests in life. There are many opportunities to get involved right here on campus. If you are interested in student government, look into running for a seat on the CUSA Council (more information at http://cusaonline.com/council.html), or try joining the New University Government (nugchair@carleton.ca). For general interests, join a CUSA club or society (http://cusaonline.com/clubs/index.html). To get some experience in the high tech community, volunteer some time for the CCSS and do office hours or help organize events (volunteer online at http://ccss.carleton.ca/).

- **Enroll in co-op.** It is well worth participating in the co-op program, so make sure your grades are always high enough to be able to. This means you need an overall CGPA of 8.0. Even though you pay extra to enroll in co-op terms, co-op jobs usually pay more than regular summer jobs. Furthermore, doing co-op is the best way to get your foot in the door at many of the big tech companies. Some employers use co-op exclusively to recruit, and others regard co-op experience more highly than other experience.

Top Five Traditionally difficult courses

5 COMP 3007 - Programming Paradigms

Not everyone will tell you that this class was a challenge. This is because you either "get it" or you just don't. If you don't, you'll realize this early on, probably during the first assignment at the latest. Do not let it slide. Because the material in this class forces students to think in a whole new way, you must get into the new kind of thinking early. See the teaching assistants and professors for help as soon as you can.

4 COMP 3804/4804 - Design and Analysis of Algorithms I/II

Though the questions given for these classes aren't long and hard to solve, finding the approach can be killer. Be prepared to spend a lot of time thinking about how to approach the problem, and when you get frustrated, take a break, and move on to the next one for a while.

3 MATH 2007 - Elementary Calculus II

While not all streams require this course, those students that do have to take it should be forewarned. Because there are often no assignments, and sometimes no tests or quizzes except the midterm and final exams, motivation to complete the practice exercises can be lacking. But for the same reasons, doing as many practice problems as possible is crucial to succeeding. Calculus isn't the easiest of the maths, and requires a lot of repetition to commit things to memory.

2 COMP 1805 - Discrete Structures

This class is the most difficult first year course because it covers such a wide variety topics that will show up in later years over and over again. To avoid having to take it again to improve your grade, be vigilant about reading the textbook, understanding the examples, and starting assignments on time. Take advantage of the tutorials that come with this class – they were added for a reason.

1 COMP 3004 - Object-Oriented Software Engineering

This is the number one course every upper year student will warn you about. Although not easy, it's not the material's difficulty level that makes this class such a challenge. It is the fact that you must work with a group many long hours to complete a semester long project. Make sure you choose reliable, hardworking group members, set up a versioning system for documentation and source files, and start working on your project early and steadily.

Java Cheat Sheet

Keywords

```
abstract boolean break byte case
catch char class continue default
do double else extends final
finally float for if implements
import instanceof int interface
long native new package private
protected public return short
static super switch synchronized
this throw throws transient try
void volatile while
```

Operators

Arithmetic `+ - * / ++ -- %`
(addition, subtraction, multiplication, division, increment, decrement, modulus

Relational `== != > < >= <=`
(equal, not equal, greater than, less than, greater than or equal, less than or equal)

Logical `& | ! ^ || &&`
(AND, OR, NOT, XOR, short circuit OR, AND)

Bitwise `& | ~ ^ >> >>> <<`
(AND, OR, NOT, XOR, shift right, shift right zero fill, shift left)

Variables

```
{public | private } [static] type
name [= expression];
```

Comments

```
// Rest of line
/* Multi-line */
/** Documentation comment */
```

Primitive Data Types

boolean	1 byte	false to true
char	2 byte	'\u0000' to '\uffff'
byte	1 byte	-128 to 127
short	2 bytes	-32768 to 32767
int	4 bytes	-2,147,483,648 to +2,147,483,647
long	8 bytes	-9,223,372,036,854,775,808 to +9,223,372,036,854,775,807
float	4 bytes	±1.40129846432481707e-45 to ±3.40282346638528860e+38
double	8 bytes	±4.94065645841246544e-324 to ±1.79769313486231570e+308

Control

```
if (Boolean-expression) statement1; [ else statement2; ]

[initialization]
while (termination-clause) {
body;
[iteration;]
}

do {
body; [iteration;]
} while (termination-clause);

for (initialization; termination-clause; iteration) {body;}

class className {
public static void main (String args[ ]) {statements;}
method definition1
…
method definitionN
}
```

Methods

```
{public | private } [static]
{type | void} name(arg, ..., arg )
{statements}
```

(continued...)

Feature

Java Cheat Sheet

Referenced Data Types

- Object assignment does not copy the object - just the reference. Use clone() to copy the object.
- The equality operator only compares references. Use equals() to compare objects.
- Declaring a reference final does not protect the object's contents if the object definition allows them to change.

Classes

- Final classes cannot be sub-classed. Final methods cannot be overridden in a subclass. Final variables are constants that cannot be changed after initialization. Final method arguments will not be changed by the method.
- Static methods or variables are not re-instantiated with every new object created; all instances share just one copy.
- Abstract classes cannot be instantiated.
- Abstract methods have no implementation and are meant to be overridden. They can only exist in abstract classes.
- Multiple inheritance is not allowed.

Interfaces

- Contain only abstract methods and final, static variables.
- Classes can implement any number of interfaces.

Arrays

- Arrays of size N are indexed by integers 0 through N-1.
- Array bounds are checked at runtime and can result in ArrayIndexOutOfBoundsException being thrown.

Resources

Java Standard Edition Downloads
http://java.sun.com/javase/downloads/index.jsp

Eclipse Download
http://www.eclipse.org/downloads/

Java API
http://java.sun.com/reference/api/

Sun Java Certification
http://www.sun.com/training/certification/java/index.html

Sun Java Tutorials
http://java.sun.com/docs/books/tutorial/

Java in a Nutshell by David Flanagan
http://www.amazon.ca/gp/product/0596007736/701-3311801-7005158?v=glance&n=916520&s=gateway&v=glance

C Cheat Sheet

Key Words

```
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
```

Operators (grouped by precedence)

`() [] . ->`
(parenthesis, brackets, structure member operator, structure pointer)

`++, --, +, -, !, ~, *pointer, &name, (type) expr, sizeof`
(increment, decrement, plus, minus, logical not, bitwise not, pointer indirection, address of object, cast expression to type, size of an object)

`* / %`
(multiply, divide, modulus)

`+ - << >> > >= < <= == != & ^`
(add, subtract, left shift, right shift, greater than, greater than or equal, less than, less than or equal, equal, not equal, bitwise and, bitwise exclusive or)

`|, &&, ||, exp1 ? exp2 : exp3, += etc, ,`
(bitwise or, logical and, logical or, conditional expression, assignment operators, expression evaluation separator)

Comments

`/* Commented line or lines */`

Input/Output

See C++ Cheat Sheet (second page)

(continued...)

Data Types / Declarations

```
char  1 byte     signed      -128 to 127
                 unsigned    0 to 255
short 2 bytes    signed      -32,768 to 32,767
                 unsigned    0 to 65,535
long  4 bytes    signed      -2,147,483,648 to
                             2,147,483,647
                 unsigned    0 to
                             4,294,967,295

int   varies depending on system

float       4 bytes    3.4E +/- 38 (7 digits)
double      8 bytes    1.7E +/- 308 (15 digits)
long double 10 bytes   1.2E +/- 4,932 (19 digits)

enum        enumeration constant
const       constant (unchanging) value
extern      declare external variable
register    register variable
static      local to source file
void        no value
struct      structure

typedef type name
            create name by data type
sizeof object
            size of an object (type is size_t)
sizeof(type name)
            size of a data type (type is size_t)
```

Functions

Functions must be prototyped before the main function, and defined after the main function.

```
type fnc (type 1, ... )
(function declarations)

main() {
  /* declarations */
  /* statements */}
(main routine with local variable declarations and
statements)
```

Feature

C Cheat Sheet

Control

`break`	exit from switch, while, do, for
`continue`	next iteration of while, do, for
`goto label`	go to
`label;`	label where goto goes
`return expr`	return value from functions

```
if (expr ) statement
else if (expr ) statement
else statement

while (expr)
  statement

for (expr 1; expr 2; expr 3)
  statement

do
while(expr );

switch (expr ) {
  case const 1 : statement 1 break;
  case const 2 : statement 2 break;
  default: statement
}
```

Preprocessor

`#include <filename>`	include library file
`#include "filename"`	include user file
`#define name text`	replacement text
`#define name(var) text`	replacement macro
`#undef name`	undefine
`#if, #else, #elif, #endif`	conditional execution
`#ifdef, #ifndef`	is name defined, not defined?
`defined(name)`	name defined?

Pointers, Arrays, Structures

`type *name`	declare pointer to type
`type *f()`	declare function returning pointer to type
`type (*pf)()`	declare pointer to function returning type
`void *`	generic pointer type
`NULL`	null pointer
`*pointer`	object pointed to by pointer
`&name`	address of object name
`name [dim]`	array
`name [dim 1][dim 2]...`	multi-dimension array
`type name[size];`	declare an array of size with data type

`type name[] = { value1, value2, ... }`
declare and initialize an array with values

`char name[] = "string";`
declare a character string array and initialize it with a string

`struct tag { /* declarations */ };`
structure template and declaration of members

`struct tag name`	create structure
`name .member`	member of structure from template
`pointer -> member`	member of pointed to structure
`union`	single value, multiple type structure
`member : b`	bit field with b bits

Resources

MSDNAA for Carleton students (download Visual Studio for free)
http://msdnaa.carleton.ca/

C Programming Tutorial
http://www.iu.hio.no/~mark/CTutorial/CTutorial.html

C Programming Language by Brian W. Kernighan and Dennis Ritchie
http://www.amazon.ca/gp/product/0131103628/701-3311801-7005158?v=glance&n=916520&s=gateway&v=glance

Feature

C++ Cheat Sheet

Key Words

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

Operators

`::`	scope

(the rest are the same as the C operators)

Comments

```
// single line comment
/* multi-line comment */
```

Data Types / Declarations

Variable declaration:
`special class size sign type name;`
- special: `volatile`
- class: `register, static, extern, auto`
- size: `long, short, double`
- sign: `signed, unsigned`
- type: `int, float, char, etc` (required)
- name: the variable name (required)

```
bool        1 byte      true or false
wchar_t     2 bytes     wide characters
```

(the rest are the same as the C data types and declarations)

Functions

```
type name(arg1, arg2, ...) {
  statement1;
  statement2;
  ...
}
```

`type`	return type of the function
`name`	name by which the function is called
`arg1, arg2,`	parameters to the function
`statement`	statements inside the function

- Functions may, but do not need to be, prototyped. C++ functions must be defined before the location where they are called from.
- Pass parameter by value: Variable is passed into the function and can be changed, but changes are not passed back. `function(int var);`
- Pass parameter by constant value: Variable is passed into the function but cannot be changed. `function(const int var);`
- Pass parameter by reference: Variable is passed into the function and can be changed, changes are passed back. `function(int &var);`
- Pass parameter by constant reference: Variable cannot be changed in the function but the data does not need to be copied into the parameter variable. `function(const int &var);`
- Pass an array by reference: It's a waste of memory to pass arrays and structures by value; instead, pass by reference. `int aryfunc(int *array[1]);`
- Default parameter values: A default parameter does not need to be passed in for the function to be called. `int add(int a, int b=2);`
- Overloading functions: Functions can have the same name, and same number of parameters as long as the parameters are of different types.

(continued...)

Feature

C++ Cheat Sheet

Console Input / Output (same as C)

`stdin` standard input stream
`stdout` standard output stream
`stderr` standard error stream

Print to screen with formatting: `printf("format", arg1,arg2,...);`
Print to strings: `sprintf(s,"format", arg1, arg2,...);`

Read data from keyboard into name1,name2,...:
`scanf("format",&name1,&name2, ...);`
Read from string s:
`sscanf("format",&name1,&name2, ...);`

Formatting:

`%d, %I` integer	`%c` single character
`%f` double (float)	`%o` octal
`%p` pointer	`%u` unsigned
`%s` char string	`%e, %E` exponential
`%x, %X` hexadecimal	`%n` number of chars written
`%g, %G` same as f for e, E	

C++ Only

`cout<<` console out, printing to screen
`cin>>` console in, reading from keyboard
`cerr<<` console error
`clog<<` console log

Dynamic Memory

`type *ptr = new type;`
allocate memory for data of type, save the ptr to it

`type *ptr = new type[size];`
allocate memory for array of size containing data of type, save the ptr to it

`delete ptr;`
deallocate memory stored in ptr

`delete [] ptr;`
deallocate memory for a whole array

Classes

```
class classname {
    public:
        member1;
   protected:
        member2;
   private:
        member3;
} objectname;

// constructor (initializes variables)
classname::classname(parms) { }

// destructor (deletes variables)
classname::~classname() {}
```

- Public members are accessible from anywhere where the class is visible
- Protected members are only accessible from members of the same class or of a friend class
- Private members are accessible from members of the same class, members of the derived classes and a friend class
- Operators and constructors can be overloaded.
- Static variables are the same throughout all instances of a class.
- A virtual function is a function member of a class, declared using the "virtual" keyword. A pointer to a derived class object may be assigned to a base class pointer, and a virtual function called through the pointer.
- The `this` keyword refers to the memory location of the current object.

Resources

C++ Reference
http://www.cplusplus.com/ref/

Effective C++ by Scott Myers
http://www.amazon.ca/gp/product/0321334876/701-3311801-7005158?v=glance&n=916520&s=gateway&v=glance

C++ Programming Language by Bjarne Stroustrup
http://www.amazon.ca/gp/product/0201700735/701-3311801-7005158?v=glance&n=916520&s=gateway&v=glance

Credits

'Computer Science Survival Guide'
written by Gail Banaszkiewicz

in collaboration with the
Carleton Computer Science Society
2006-2007 Executive

Designed by Gail Banaszkiewicz

Photos by Christine Wang and Peter Hoang

2006-2007 Executive

Gail Banaszkiewicz (President)
Christian Muise (Communications Liaison, Nexus)
Anne Taylor (Treasurer)
Geoff Foster (General Exec)
Elan Dubrofsky (General Exec)
Nicholas Bale (General Exec)
Suhkveer Matharu (General Exec)