A Cookie Love Story
Last-Minute Shopping

If you're like some of us here at Flyby, you may have extended your Labor Day weekend a little too far—both before and after. But even if you managed to skip all of Shopping Period until today, you definitely still have some options (and time) to fill up your study card.

Science of the Physical Universe 20: What is Life? From Quarks to Consciousness—This Gen Ed course also fulfills either Science A or B for the Core, so it should probably be a pretty flexible option for most of you. Plus, even though the class has met two times already, we know from experience that most of your learning will take place in section. So just make sure you show up to that!

Ethical Reasoning 22: Justice—This ever-popular Sandel class met once last week, but it's not like they'll know you weren't there.

Computer Science 50: Introduction to Computer Science I—Missed the first lecture? Don't worry, it's already online, so you can catch up for the second class easily. And now that you can take it pass/fail and as a Gen Ed or Core course, it's more attractive than ever.

English 62: Diffusions: Castaways and Renegades—Double colon in the title aside, this course doesn't seem too complicated. The first lecture, according to the syllabus, only included an introduction and an overview. Read classics by D. H. Lawrence, Stowe, Whitman, Fitzgerald and Twain (you know you've always meant to). Added bonus? No midterm, no final exam.

History and Literature 90: Stories of Slavery and Freedom—This course meets for the first time this week, and as far as we can tell, you don't need to have prepared anything in advance. Do brace yourself for a possible crowd, however—Professor Timothy P. McCarthy is popular and the Q guide ratings are stellar.

Science of Living Systems 11: Molecules of Life—Want an easy way to kill off a Gen Ed or Core requirement? This class is your answer (for Core people, it fulfills Science B). Professor David R. Liu and Professor Jon Clardy are sure to be a good time in this large and popular course about the small and unseen.

Photo courtesy of Wikimedia Commons.
office hours
sections
when clicked

say O hai, world!
```c
int main()
{
    printf("O hai, world!\n");
}
```
statements
statements

printf("O hai, world!\n");
loops
loops

while (true) {
    printf("O hai!\n");
}
loops
for (int i = 0; i < 10; i++)
{
    printf("O hai!\n");
}
variables
variables

```c
int counter = 0;
while (true)
{
    printf("%d\n", counter);
    counter++;
}
```
Boolean expressions

$x < y \text{ and } y < z$
Boolean expressions

```
(x < y)
((x < y) && (y < z))
```
conditions
conditions

```plaintext
if (x < y)
{
    printf("x is less than y\n");
}
else if (x > y)
{
    printf("x is greater than y\n");
}
else
{
    printf("x is equal to y\n");
}
```
arrays

```c
string inventory[1];
inventory[0] = "Orange";
```
#include <stdio.h>

int
main()
{
    printf("O hai, world!\n");
}

how to write a program

1. nano hello.c
2. gcc hello.c
3. ./a.out
how to write a program

1. nano hello.c
2. gcc -o hello hello.c
3. ./hello
how to write a program

1. nano hello.c
2. make hello
3. ./hello
functions

main ...
Standard Library

printf

...
CS50 Library

GetChar
GetDouble
GetFloat
GetInt
GetLongLong
GetString
printf

%c %d %f %lld %s ...
escape sequences
\n \r \t \' \" \\ \0 ...
math
+
-
*
/
%
primitive types

char  double  float  int  long  long  ...
CS50 types

bool  string  ...
<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
<th>Associativity</th>
</tr>
</thead>
<tbody>
<tr>
<td>( )</td>
<td>Parentheses (grouping)</td>
<td>left-to-right</td>
</tr>
<tr>
<td>[ ]</td>
<td>Brackets (array subscript)</td>
<td>left-to-right</td>
</tr>
<tr>
<td>.</td>
<td>Member selection via object name</td>
<td>left-to-right</td>
</tr>
<tr>
<td>-&gt;</td>
<td>Member selection via pointer</td>
<td>left-to-right</td>
</tr>
<tr>
<td>++ --</td>
<td>Prefix increment/decrement (see Note 1)</td>
<td>right-to-left</td>
</tr>
<tr>
<td>++</td>
<td>Prefix increment</td>
<td>right-to-left</td>
</tr>
<tr>
<td>-</td>
<td>Unary minus</td>
<td>left-to-right</td>
</tr>
<tr>
<td>! ~</td>
<td>Logical negation/bitwise complement</td>
<td>left-to-right</td>
</tr>
<tr>
<td>(type)</td>
<td>Cast (change type)</td>
<td>left-to-right</td>
</tr>
<tr>
<td>.</td>
<td>Dereference</td>
<td>left-to-right</td>
</tr>
<tr>
<td>&amp;</td>
<td>Address</td>
<td>left-to-right</td>
</tr>
<tr>
<td>sizeof</td>
<td>Determine size in bytes</td>
<td>left-to-right</td>
</tr>
<tr>
<td>* / %</td>
<td>Multiplication/division/modulus</td>
<td>left-to-right</td>
</tr>
<tr>
<td>+ -</td>
<td>Addition/subtraction</td>
<td>left-to-right</td>
</tr>
<tr>
<td>&lt;&lt; &gt;&gt;</td>
<td>Bitwise shift left, Bitwise shift right</td>
<td>left-to-right</td>
</tr>
<tr>
<td>&lt; &lt;= &gt; &gt;=</td>
<td>Relational less than/less than or equal to, Relational greater than/greater than or equal to</td>
<td>left-to-right</td>
</tr>
<tr>
<td>== !=</td>
<td>Relational is equal/to be not equal to</td>
<td>left-to-right</td>
</tr>
<tr>
<td>&amp;</td>
<td>Bitwise AND</td>
<td>left-to-right</td>
</tr>
<tr>
<td>^</td>
<td>Bitwise exclusive OR</td>
<td>left-to-right</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Bitwise inclusive OR</td>
</tr>
<tr>
<td>&amp; &amp;</td>
<td>Logical AND</td>
<td>left-to-right</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Logical OR</td>
</tr>
<tr>
<td>? :</td>
<td>Ternary conditional</td>
<td>right-to-left</td>
</tr>
<tr>
<td>=</td>
<td>Assignment</td>
<td>right-to-left</td>
</tr>
<tr>
<td>+= -=</td>
<td>Addition/subtraction assignment</td>
<td>right-to-left</td>
</tr>
<tr>
<td>*= /=</td>
<td>Multiplication/division assignment</td>
<td>right-to-left</td>
</tr>
<tr>
<td>%= %&amp;</td>
<td>Modulo/bitwise AND assignment</td>
<td>right-to-left</td>
</tr>
<tr>
<td>^=</td>
<td>=</td>
<td>Bitwise exclusive OR assignment</td>
</tr>
<tr>
<td>&lt;&lt;= &gt;&gt;=</td>
<td>Bitwise shift left/right assignment</td>
<td>right-to-left</td>
</tr>
<tr>
<td>,</td>
<td>Comma (separate expressions)</td>
<td>left-to-right</td>
</tr>
</tbody>
</table>
how to write a program

1. nano hello.c
2. gcc -o hello hello -lcs50
3. ./hello
to be continued...