"Cheat Sheet" - Week 1

CS50 — Fall 2010

Prepared by: Doug Lloyd '09

September 20, 2010

Types

C's built-in types (at least the ones we'll be using most are):

```
char
int
float
double
void // Not for variables, though!
```

The two extra types, CS50 types, that you get by having `#include <cs50.h>` atop your .c files and linking with the `-lcs50` flag are:

```
string
bool
```

Comments

Comments can be done in one of two ways. You can have "inline" comments, which typically span only one line and start with `//`:

```
// this loop counts to 10
for(int i = 1; i <= 10; i++) {
    printf("%d, ah ah ah..."); // like the Count from Sesame Street
    sleep(1);
}
```

You can also have "block" comments, which span multiple lines. These start with `/*` and end with `*/`. You might see these introducing a function to identify its author and purpose, or just for a complicated comment.

```
/**************************************
 * hello.c                           *
 *                                   *
 * Says "hello, world!" awesomely    *
 **************************************/
```

Using `printf()`

When using `printf()`, the following are the most common formats[1]. Don't forget that `\n` will insert a newline for you!

```
int four = 4;
char eks = 'X';
float pointfivefour = 0.54;
double pointsixeight = 0.68;
string hi = "Hello";
long long trillion = 1000000000000

printf("%d\n", four); // prints 4
printf("%c\n", eks); // prints X
printf("%.3f\n", pointfivefour); // prints (approximately) 0.54 to three decimal places
printf("%.5f\n", pointsixeight); // prints (approximately) 0.68 to five decimal places
printf("%s\n", hi); // prints Hello
printf("%lld\n", trillion); // prints 1000000000000
```

Loops

Loops come in three flavors: `while`, `do-while`, and `for`. Typically, a `while` loop is used to keep repeating the block of code within its braces until a given condition is met, but it isn't necessarily certain when that condition will be met. If the condition is met before the first iteration of the loop begins, the loop will not even run one time. To ensure that a loop runs at least once, use a `do-while`. A `for` loop is usually used to run the block within its braces a particular number of times. All three, however, can be used in most cases with equal effect. For example, all three of these loops print out the numbers 1 through 10 in a line:

```
int i = 1;                  int i = 0;                  for(int i = 1; i <= 10; i++) {
while(i <= 10) {            do {                            printf("%d ", i);
   printf("%d ", i);           i++;                     }
   i++;                         printf("%d ", i);
}                           } while(i <= 10);
```

Conditions

There are four conditional statements you are likely to encounter: `if`, `else`, `switch`, and (more rarely) `?:`. An example of each appears below:

```
int i = 6;                  int i = 6;                  int i = 6;
if(i == 5)                  switch(i) {                 j = (i == 6) ? 7 : 5;
  // take action one           case 1: case 2:          /* j will be 7 if the
else if(i == 6) {                 // take action one       condition (i == 6) is
  // take action two             break;                    true. Otherwise, j
  // print to the user        case 4:                      will be 5 */
}                                 // take action two
else                             break;
  // take action three        default:
                                  // take action three
                            }
```

---

[1]For lots more on `printf()`, check out *http://www.cplusplus.com/reference/clibrary/cstdio/printf/*