

15-122: Principles of Imperative Computation

Recitation 2 Solutions

Josh Zimmerman

Bit manipulation

Let's look at some examples of masking so you can get a better idea of how it's used. First, let's write a function that, given a pixel in the ARGB format, returns the green and blue components of it. Your solution should use only `&`.

Solution:

```
1 typedef int pixel;
2 int greenAndBlue(pixel p)
3 //@ensures 0 <= \result && \result <= 0xffff;
4 {
5     // We only want the lower 16 bits of p, so and the others with 0
6     // to get rid of them
7     return p & 0xffff;
8 }
```

Now, let's write a function that gets the alpha and red pixels of a pixel in the ARGB format. Your solution can use any of the bitwise operators, but will not need all of them.

Solution:

```
1 typedef int pixel;
2 int alphaAndRed(pixel p)
3 //@ensures 0 <= \result && \result <= 0xffff;
4 {
5     // First, we want to put the top 16 bits in the bottom of the number.
6     // Then, we want to get rid of any sign extension that the right shift
7     // caused, so we use a mask to get rid of anything above the bottom 16
8     // bits
9     return (x >> 16) & 0xffff;
10 }
```