

COMP1511 17s2

— Lecture 7 —

Array of Sunshine

Andrew Bennett

`<andrew.bennett@unsw.edu.au>`

loops and repetition
arrays of data

Admin

1

Don't panic!

assignment 0

Loops

What if we want to do something multiple times?

Use a loop!

*keep doing this **while** this condition is true*

Anatomy of a Loop

initialisation

•

condition

•

statements

•

update

•

Anatomy of a Loop

initialisation

set up our variables

condition

.

statements

.

update

.

Anatomy of a Loop

initialisation

set up our variables

condition

while “something”...

statements

.

update

.

Anatomy of a Loop

initialisation

set up our variables

condition

while “something”...

statements

things we do inside our loop

update

.

Anatomy of a Loop

initialisation

set up our variables

condition

while “something”...

statements

things we do inside our loop

update

move along to the next iteration

Aside: Definitions

iterate

perform repeatedly

iteration

the repetition of a process

A Counting Loop

“Do this thing n different times”

sometimes, it's explicit:

e.g. print out 'hello world!' 10 times

sometimes, it's not:

e.g. print out the numbers from 1-10

e.g. calculate the power of a number (*e.g.*, 2^3)

A Counting Loop

“Do this thing n times”

use a **loop counter**

... a variable that we use in our loop
to count how many times we've done something

A Counting Loop

Do something until we've done it n times
e.g. print out 'hello world!' 10 times

counter starts at 0

print "hello world!"; increase counter to 1 (we've done it once)
print "hello world!"; increase counter to 2 (we've done it twice)
print "hello world!"; increase counter to 3 (we've done it three times)

...

print "hello world!"; increase counter to 9 (we've done it 9 times)
print "hello world!"; increase counter to 10 (we've done it 10 times)

now stop, because we've done it 10 times.

A Counting Loop

How would we code this?

start our counter at 0

print "hello world!"

while counter is less than 10,
increase our counter by 1

Anatomy of a Loop

initialisation

set up our variables

condition

while “something”...

statements

things we do inside our loop

update

move along to the next iteration

```
????  
while (?????) {  
    ????  
    ????  
}
```

initialisation

```
// set up our loop counter, start at 0
while (????) {
    ???
    ???
}
```


initialisation
condition

```
// set up our loop counter, start at 0
while (something) {
    ???
    ???
}
```

initialisation
condition
statements

```
// set up our loop counter, start at 0  
while (something) {  
    // do something  
    ???  
}
```

initialisation
condition
statements
update

```
// set up our loop counter, start at 0  
while (something) {  
    // do something  
    // move to the next iteration of the loop  
}
```

initialisation
condition
statements
update

```
int i = 0;  
while (something) {  
    // do something  
    // move to the next iteration of the loop  
}
```

initialisation
condition
statements
update

```
int i = 0;  
while (i < 10) {  
    // do something  
    // move to the next iteration of the loop  
}
```

initialisation
condition
statements
update

```
int i = 0;  
while (i < 10) {  
    printf ("hello, world!\n");  
    // move to the next iteration of the loop  
}
```

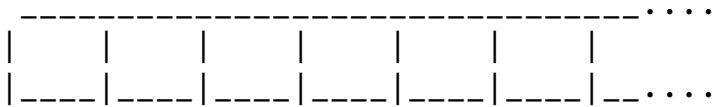
initialisation
condition
statements
update

```
int i = 0;  
while (i < 10) {  
    printf ("hello, world!\n");  
    i = i + 1;  
}
```

memory

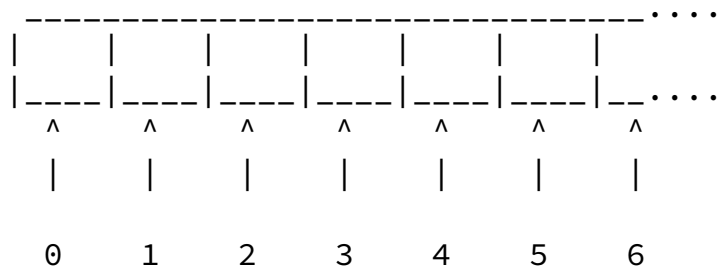
Memory is...

a series of boxes



Memory is...

a series of boxes
with addresses

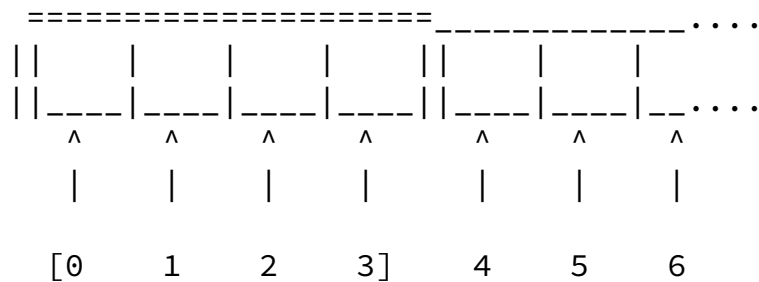


Each box represents one byte (8 bits)

Bits, Bytes, Types

Groups of boxes together form a *type*

e.g. an int is 4 bytes



Aside: Bits and Bytes

1 byte = 8 bits

each bit can be 0 or 1

how many different values can we represent in 8 bits?

Aside: Bits and Bytes

1 bit:

0, 1 (2 different values)

2 bits:

00, 01, 10, 11

0, 1, 2, 3 (4 different values)

3 bits:

000, 001, 010, 011, 100, 101, 110, 111

0, 1, 2, 3, 4, 5, 6, 7 (8 different values)

8 bits:

00000000, 00000001, 00000010, 00000011, ..., 11111111

0, 1, 2, 3, ..., ???

Aside, Aside: Converting Binary to Decimal

$$\begin{array}{c} 11111111 \\ 1 + 2 + 4 + 8 + 16 + \dots + 128 = 255 \end{array}$$

$$\begin{array}{l} 11111111 + 1 \\ = 100000000 \\ = 256 \end{array}$$

$$\begin{array}{l} 100000000 - 1 \\ = 11111111 \\ = 256 - 1 \\ = 255 \end{array}$$