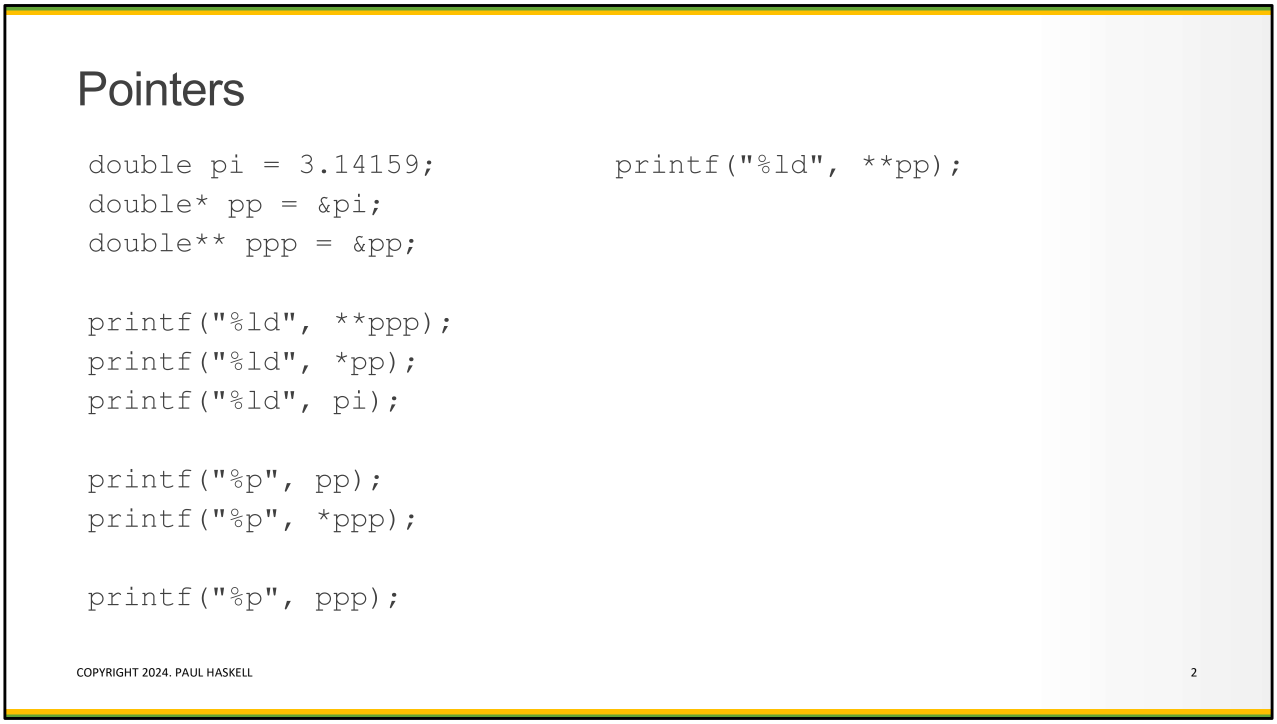
CS221
C and Systems Programming

Practice…

Demos, midterm practice, in-class lab



Pointers

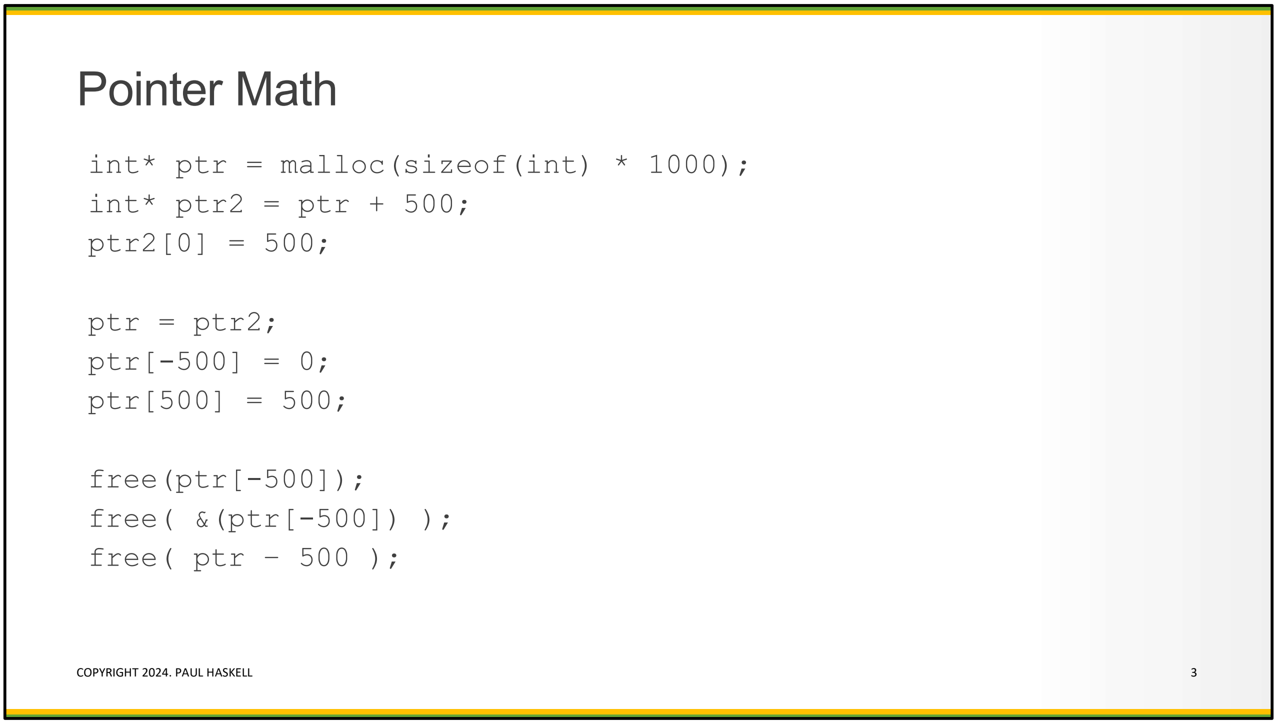
```
double pi = 3.14159;             printf("%ld", **pp);
double* pp = &pi;
double** ppp = &pp;

printf("%ld", **ppp);
printf("%ld", *pp);
printf("%ld", pi);

printf("%p", pp);
printf("%p", *ppp);

printf("%p", ppp);
```

DRAW THE PICTURES: type, variable name, value, address



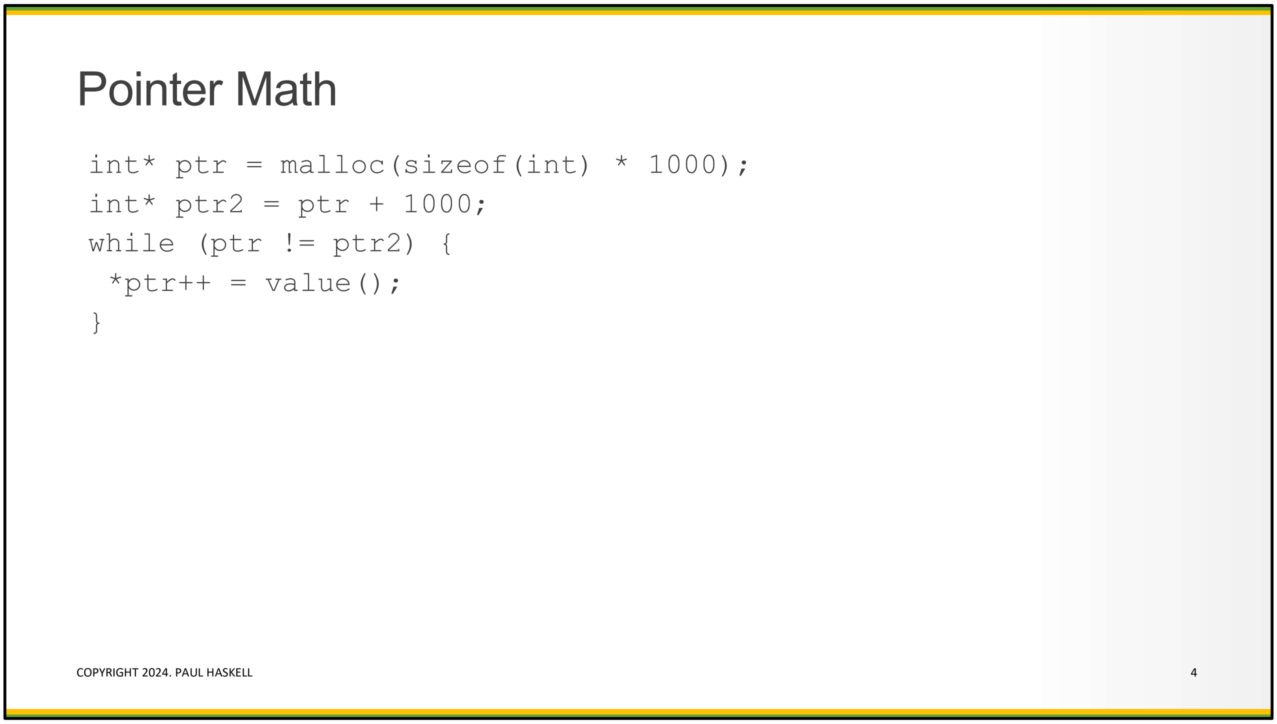
Pointer Math

```
int* ptr = malloc(sizeof(int) * 1000);
int* ptr2 = ptr + 500;
ptr2[0] = 500;

ptr = ptr2;
ptr[-500] = 0;
ptr[500] = 500;

free(ptr[-500]);
free( &(ptr[-500]) );
free( ptr - 500 );
```

DRAW THE PICTURE: type, name, value, address
Show malloc'd memory as a blob in the heap; show stack vars in the stack

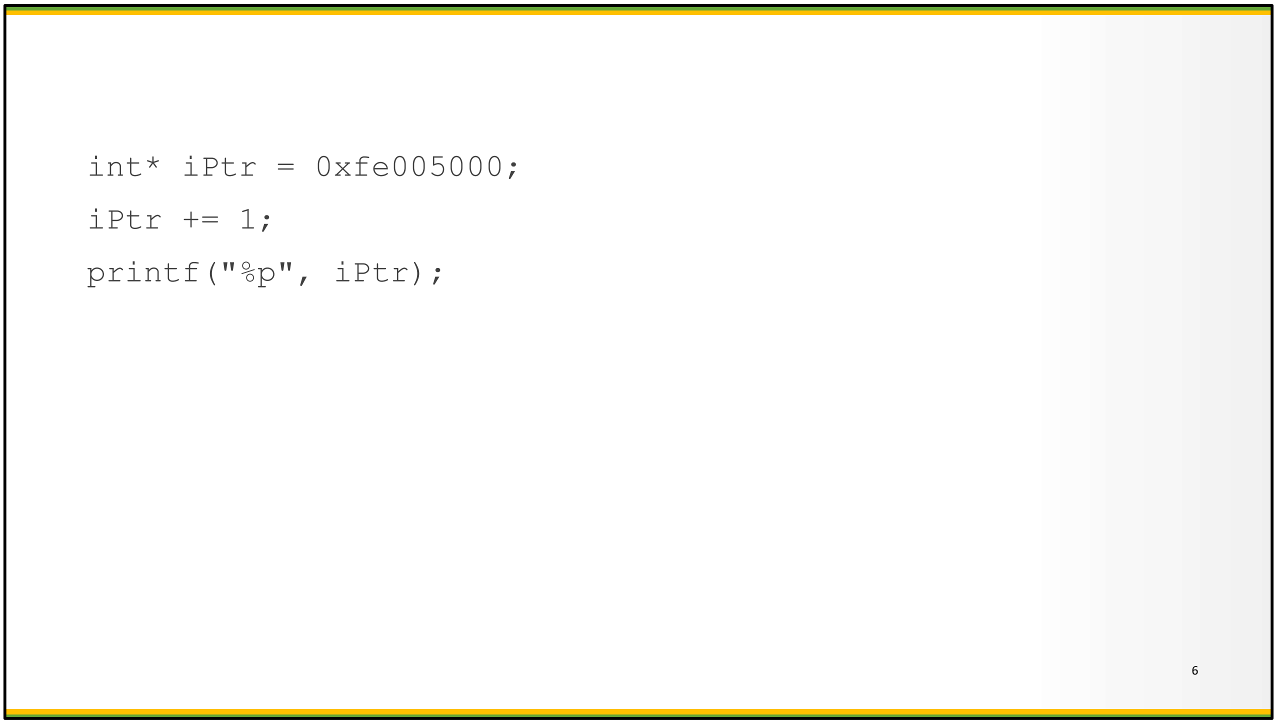


Pointer Math

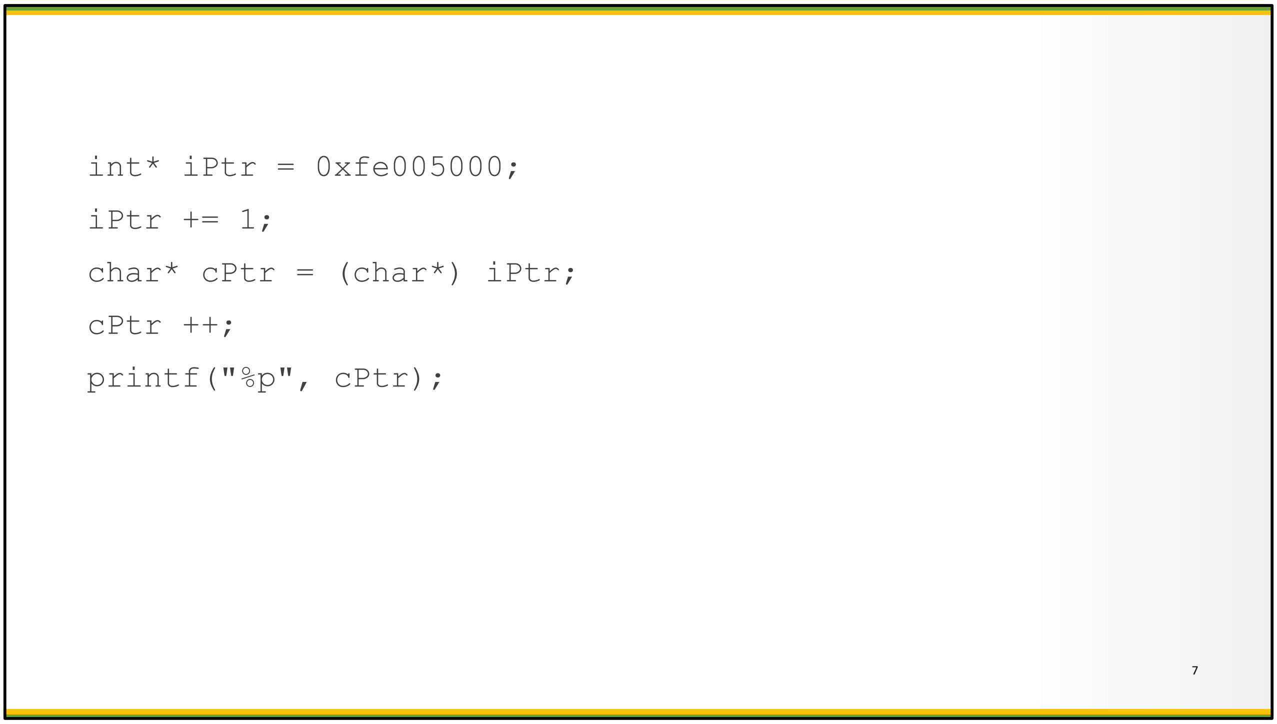
```
int* ptr = malloc(sizeof(int) * 1000);
int* ptr2 = ptr + 1000;
while (ptr != ptr2) {
 *ptr++ = value();
}
```

ptr2 is a valid pointer. We cannot read or write its location, but we can check its value.

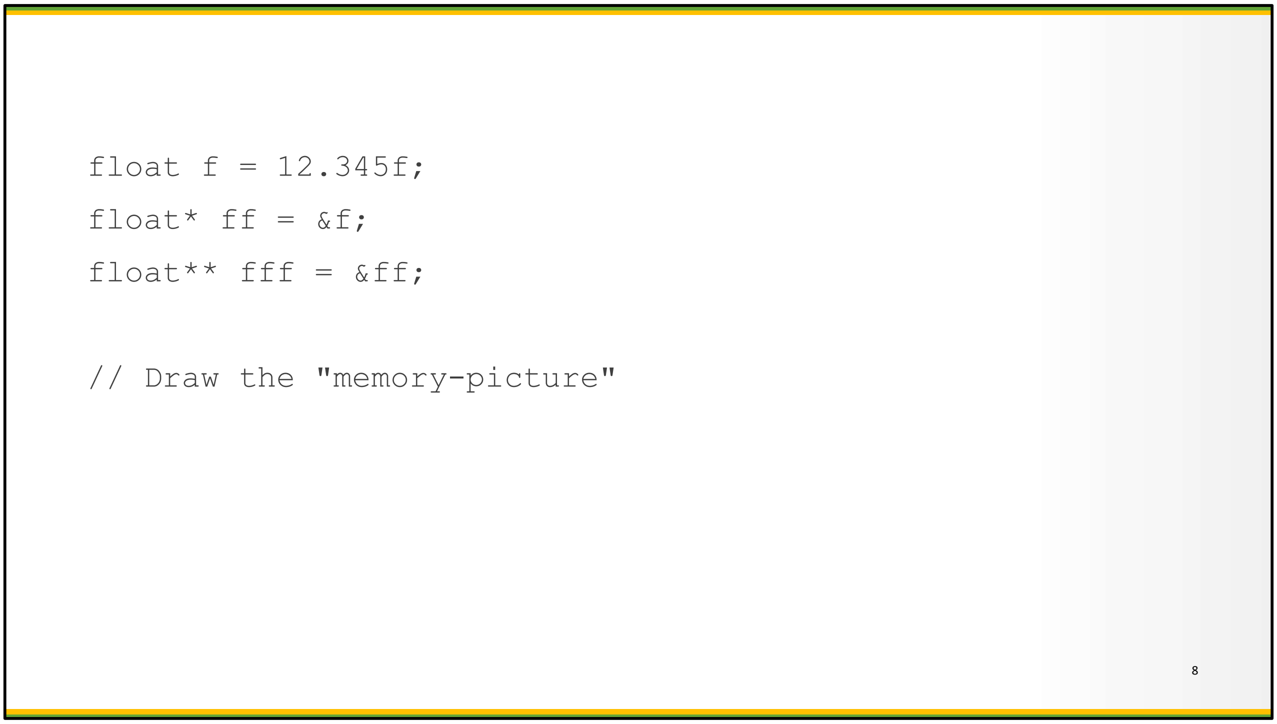
Ok, the contest



```
int* iPtr = 0xfe005000;
iPtr += 1;
printf("%p", iPtr);
```

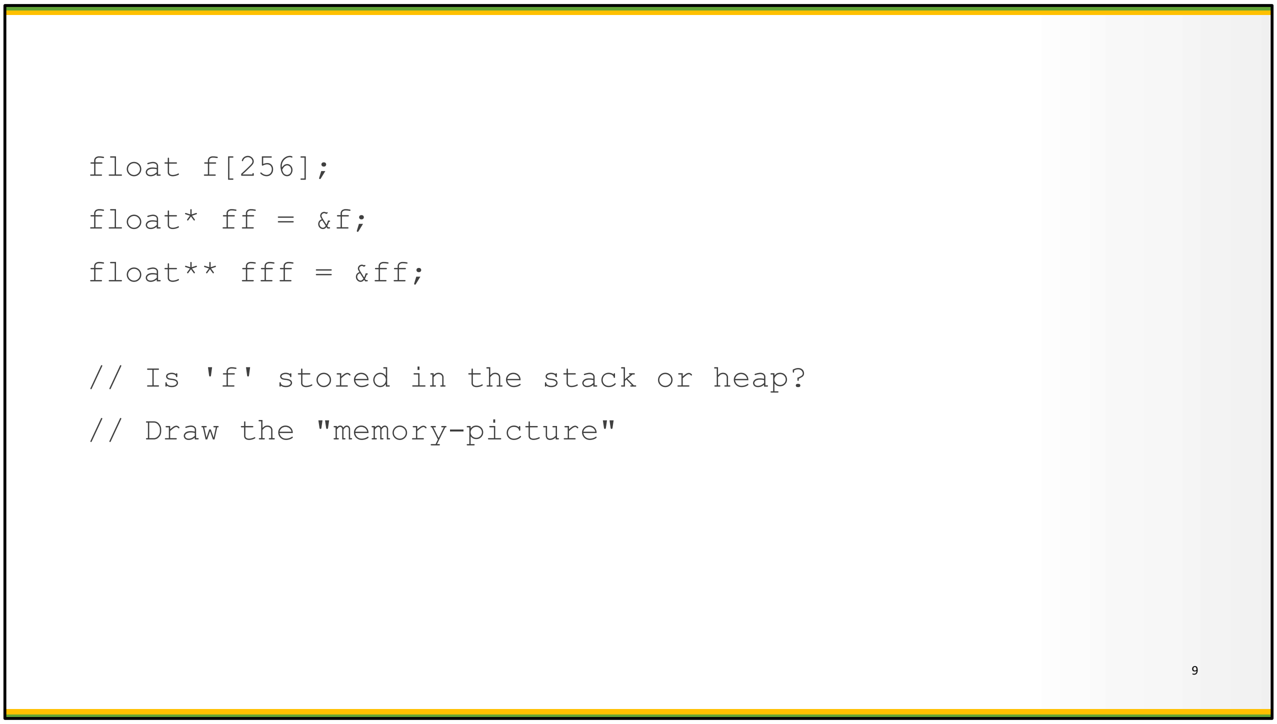


```
int* iPtr = 0xfe005000;
iPtr += 1;
char* cPtr = (char*) iPtr;
cPtr ++;
printf("%p", cPtr);
```



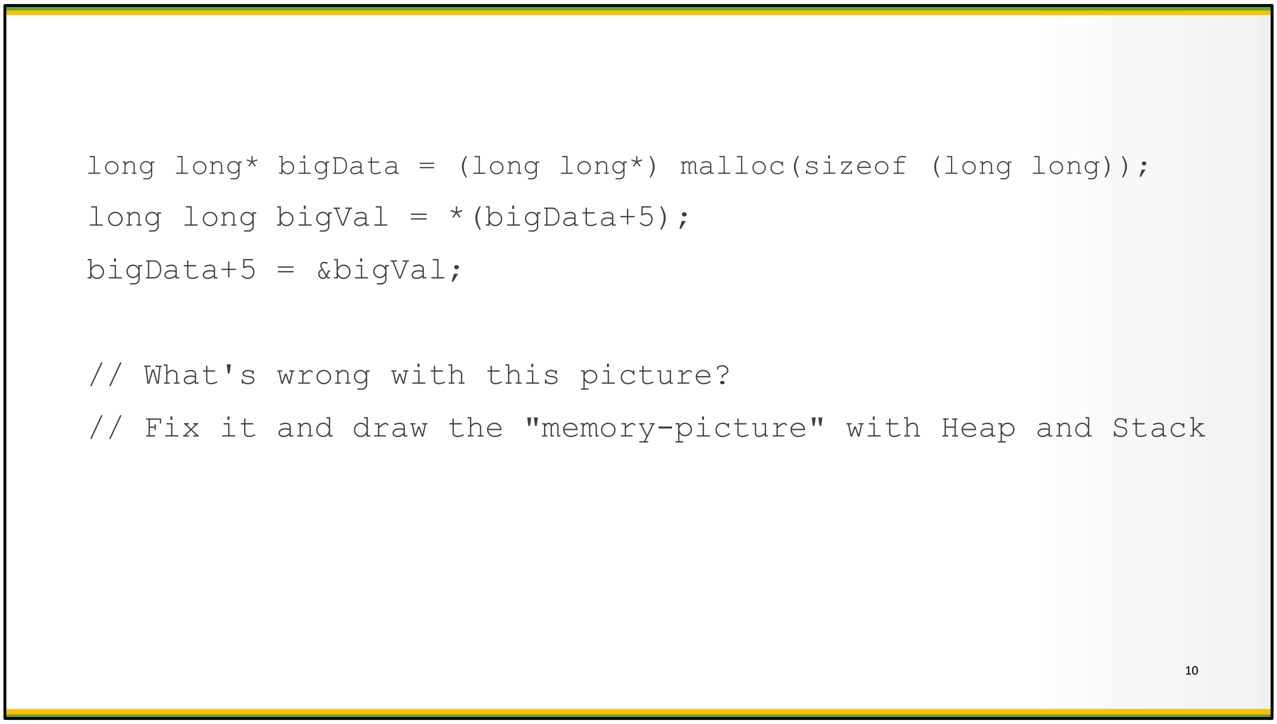
```
float f = 12.345f;
float* ff = &f;
float** fff = &ff;

// Draw the "memory-picture"
```



```
float f[256];
float* ff = &f;
float** fff = &ff;

// Is 'f' stored in the stack or heap?
// Draw the "memory-picture"
```

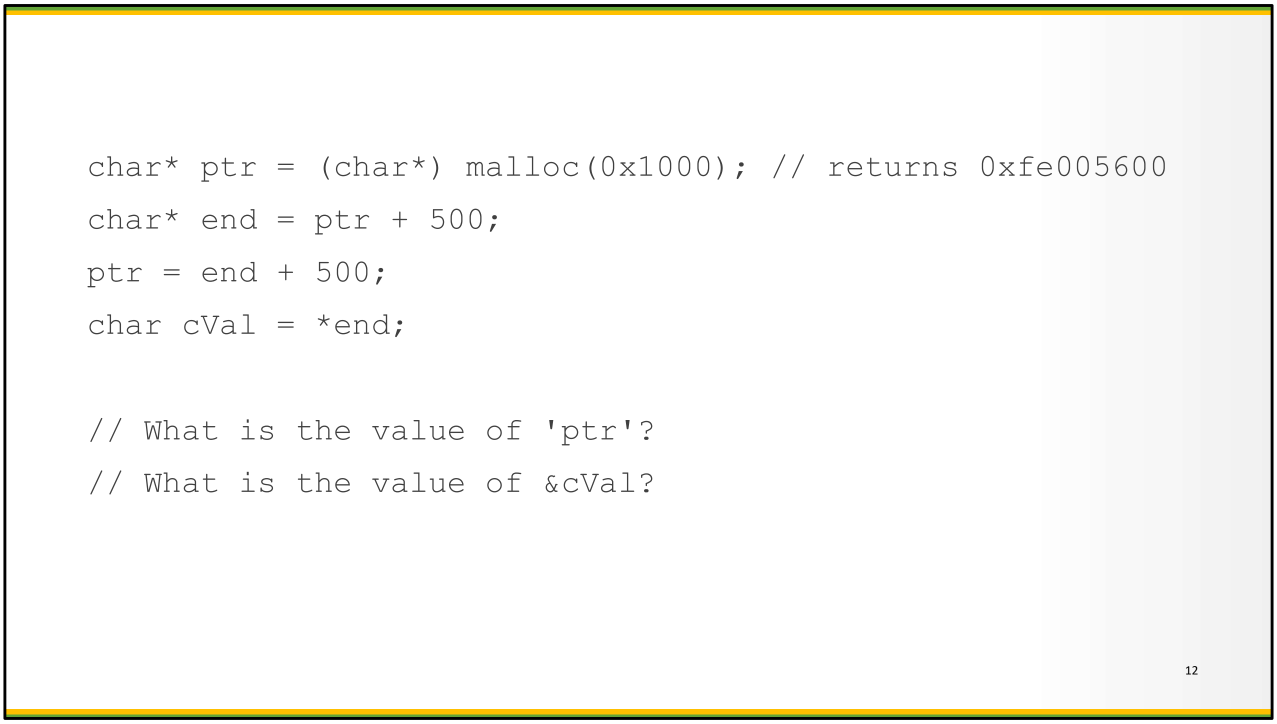


```
long long* bigData = (long long*) malloc(sizeof (long long));
long long bigVal = *(bigData+5);
bigData+5 = &bigVal;

// What's wrong with this picture?
// Fix it and draw the "memory-picture" with Heap and Stack
```

Not an l-value

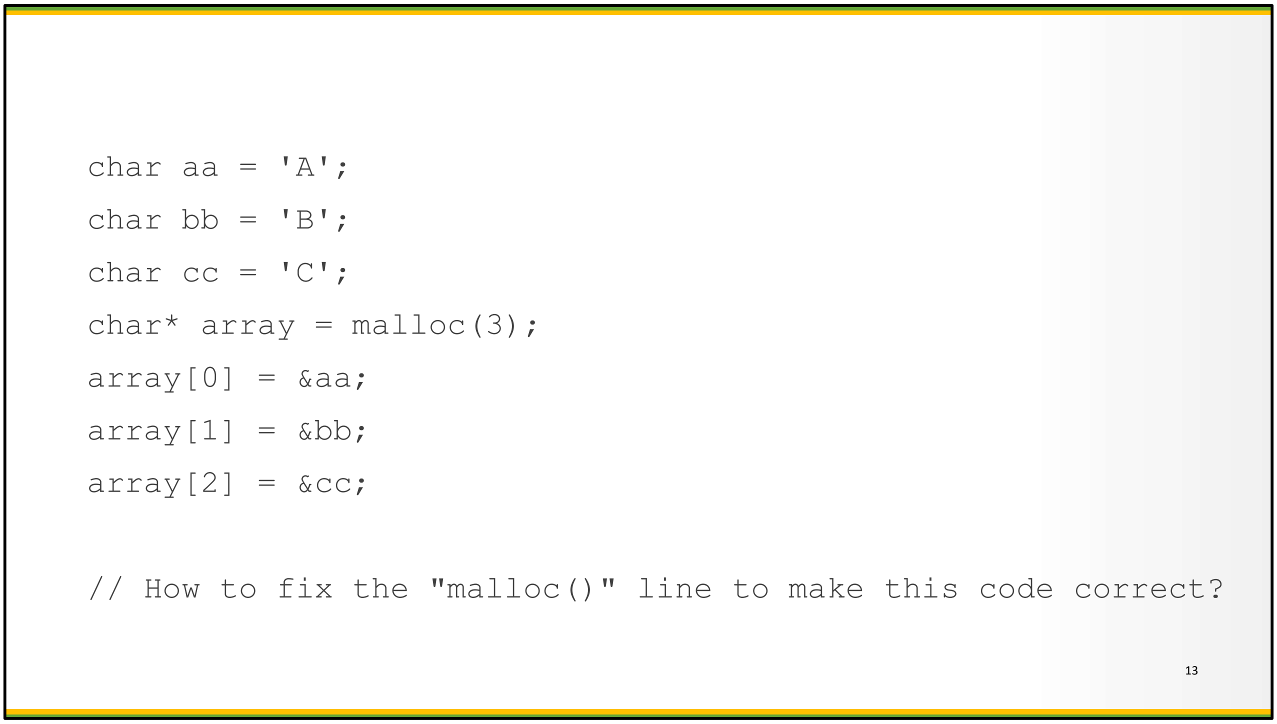
```
unsigned short val = 0x1234;
// Write the code to endian-reverse 'val'
```



```
char* ptr = (char*) malloc(0x1000); // returns 0xfe005600
char* end = ptr + 500;
ptr = end + 500;
char cVal = *end;

// What is the value of 'ptr'?
// What is the value of &cVal?
```

Draw the picture!



```
char aa = 'A';
char bb = 'B';
char cc = 'C';
char* array = malloc(3);
array[0] = &aa;
array[1] = &bb;
array[2] = &cc;

// How to fix the "malloc()" line to make this code correct?
```