



COMP1511

Tutorial 7

pointers | struct pointers | malloc

pointers





Pointer Operations

`int *ptr` - Declare an integer pointer called `ptr`

`&num` - Give me the address of variable `num`

`*ptr` - Give me the variable at the address stored in `ptr`
(dereferencing)



Pointers: declaring vs dereferencing

The asterisk (*) is used for 2 key pointer operations:

Declaring, e.g. `int *ptr;`

Dereferencing, e.g. `*ptr = 5;`

Rule of thumb: if the asterisk has a variable type before it (e.g. `int *`, `char *`) it's a pointer declaration, otherwise it's dereferencing a pointer

struct pointers





Do I use `.` or `->` ?

With a regular struct variable (e.g. a `struct person` named `student1`), we would use the `.` (dot) operator to access a field (e.g. `student1.name`)

If we had a pointer to a struct, we would need to dereference the pointer to get to the struct and then access its field: `(*student1_pointer).name`

We use the `->` operator to make this neater: `student1_pointer->name`