University of Illinois at Urbana-Champaign
Dept. of Electrical and Computer Engineering

ECE 220: Computer Systems & Programming

Dynamic Allocation in C++:
New and Delete

Avoid C's Dynamic Allocation Routines in C++

`malloc` does not call constructors.

`free` does not call destructors.

Do not use these functions to allocate or deallocate class instances!

In general,
- **when writing C++** code,
- it's best to **avoid using `malloc` and `free`** directly at all.

Use **new** to Create New Instances

To create a new object, write

```
MyClass* m = new MyClass
                (arg1, arg2, ...);
```

Returns pointer to a constructed **`MyClass`** instance (a **`MyClass*`**).

arguments passed to constructor; if omitted (no parentheses), no arguments passed

new Throws an Exception on Failure

On failure,
- **new throws an exception**,* which (by default)
- **terminates your program**.

If you want failure to return NULL
(no constructor is called in that case), use

```
#include <new>
MyClass* m = new (std::nothrow)
        MyClass (arg1, arg2, ...);
```

*We will not have time to discuss exception handling in our class.