

# 601.220 Intermediate Programming

C++ dynamic memory allocation

# C++ dynamic memory allocation

`new` and `delete` are essentially the C++ versions of `malloc` and `free`

Big differences: `new` not only allocates the memory, it also **calls the appropriate constructor** if used on a class type (more on this later)

Small differences: `new` and `delete` are **keywords** rather than functions, so you don't use `(...)` when calling them

## new usage

```
// dynamic1.cpp
1  #include <iostream>
2
3  int main() {
4      int *iptr = new int;
5      *iptr = 10;
6      std::cout << "value of iptr " << iptr << std::endl;
7      std::cout << "value in *iptr " << *iptr << std::endl;
8      return 0;
9  }

$ g++ -o dynamic1 dynamic1.cpp -std=c++11 -pedantic -Wall -Wextra
$ ./dynamic1

value of iptr 0xd2ac20
value in *iptr 10
```

## delete usage

`delete` deletes something allocated with `new`

*// dynamic2.cpp*

```
1  #include <iostream>
2  int main() {
3      int *iptr = new int;
4      *iptr = 10;
5      // do more with iptr
6      delete iptr;
7      std::cout << "after delete" << std::endl;
8      std::cout << "value in *iptr " << *iptr << std::endl;
9      std::cout << "value of iptr " << iptr << std::endl;
10     // note: new and delete don't use parentheses,
11     // unlike malloc() / free()
12     return 0;
13 }
```

## delete usage

```
$ g++ -o dynamic2 dynamic2.cpp -std=c++11 -pedantic -Wall -Wextra
```

```
$ ./dynamic2
```

after delete

value in \*iptr 0

value of iptr 0x1a46c20

# C++ dynamic array allocation

`T * fresh = new T[n]` allocates an array of `n` elements of type `T`

Use `delete[] fresh` to deallocate – always use `delete[]` (not `delete`) to deallocate a pointer returned by `new T[n]`

If `T` is a **built-in** type (`int`, `float`, `char`, etc), then the values are *not initialized*, like with `malloc`

If `T` is a `class`, then `T`'s default constructor is called for **each** element allocated (more on this soon)

## C++ dynamic array allocation in action

```
// dynamic3.cpp
```

```
1  #include <iostream>
2
3  int main() {
4      double *d_array = new double[10];
5      for(int i = 0; i < 10; i++) {
6          std::cout << (d_array[i] = i * 2) << " ";
7      }
8      std::cout << std::endl;
9      delete[] d_array;
10     return 0;
11 }
```

```
$ g++ -o dynamic3 dynamic3.cpp -std=c++11 -pedantic -Wall -Wextra
$ ./dynamic3
0 2 4 6 8 10 12 14 16 18
```