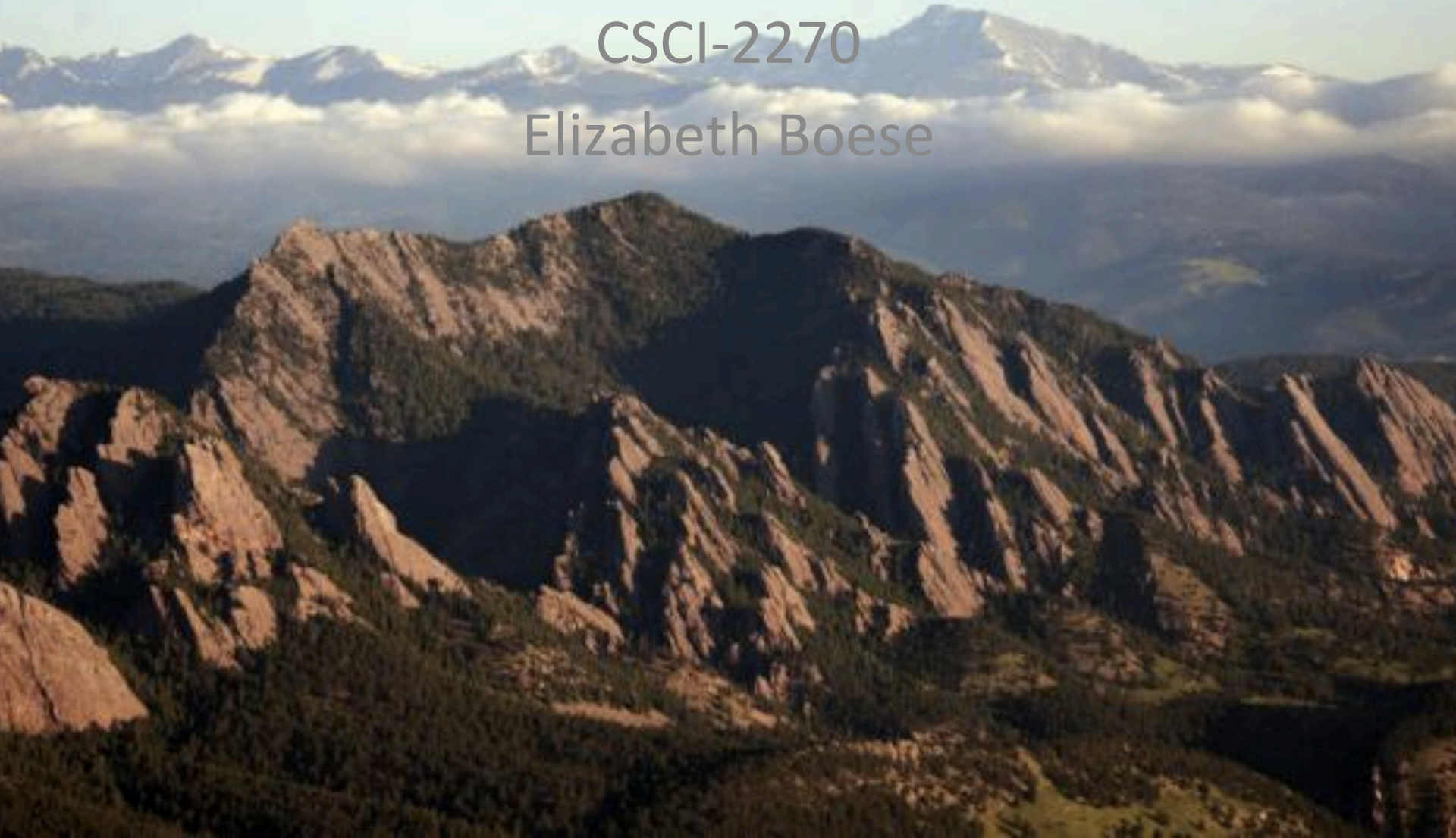
Dynamic Memory



CSCI-2270

Elizabeth Boese

Dynamic Variables

- **Dynamic variable –** memory allocated to a variable during program execution
- **Pointers** are used to create dynamic variables.
- The memory size pointed to by a pointer can be created and destroyed by two operators
  – **`new`** and **`delete`**
- **`new`** and **`delete`** are two reserved words in C++

Syntax to use operator new

- Syntax:
  - single variable
    ```
    new dataType;
    ```
  - array
    ```
    new dataType[intExp];
    ```
- Examples:

  To allocate space for an integer and point pointer p to it:

  ```
  int *p;
  p = new int;
  ```

  To allocate space for a char array and point pointer q to it:

  ```
  char *q;
  q = new char[80];
  ```

Syntax to use operator delete

- Syntax:
  - to destroy a single dynamic variable

    ```
    delete pointer;
    ```
  - to destroy a dynamically created array of variables

    ```
    delete [] pointer;
    ```

- Examples:

  To return the memory dynamically allocated to p and q on the previous slide:

  ```
  delete p;
  delete [] q;
  ```

delete operator

- Note: you can only “delete” memory that you have allocated using “new.”

delete operator

- Note: *delete* does not actually remove the pointer or the variable being pointed to from memory.
- In fact, the pointer will still point to the variable, which still resides in memory.
- The *delete* command simply tells the system that if it needs memory, it can re-use the location.
- It is up to the programmer to avoid de-referencing pointers that have been deleted.

Dynamic Memory Allocation

- Tip: after calling delete, set the pointer to NULL:

```
delete myPtr;
myPtr = NULL;
```

- If you "new" any memory, make sure you "delete" it as soon as you are finished with it.

struct Example

```
struct Student
{
    string firstName, lastName, aNumber;
    double GPA;
};
int main()
{
   Student* student1 = new Student;
   …
   delete student1;
}
```

Class Example

```
ComplexNum *pz = new ComplexNum(3, 4);
```

The new operator returns the address of the first byte of the memory allocated

Class Destructors

Any time a class allocates a system resource…

Reserves memory using the *new* command

Opens a disk file

Connects to another computer over the network

Your class must have a destructor that…

Frees the allocated memory with the *delete* command

Closes the disk file

Disconnects from the other computer

Class Destructors

```
class SomeClass
{
 public:
    ~SomeClass();
    …
};

SomeClass::~SomeClass()
{
}
```