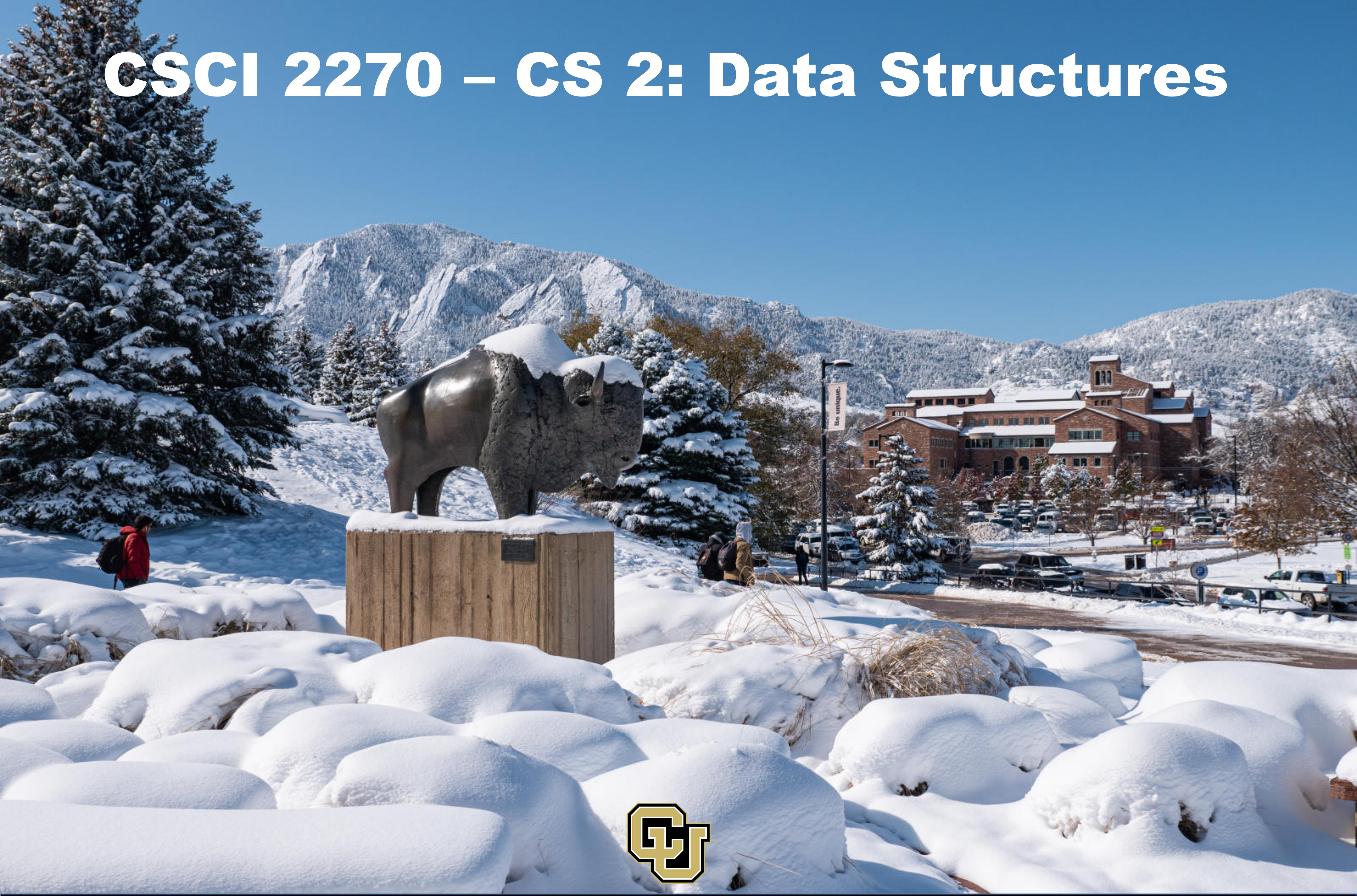
CSCI 2270 – CS 2: Data Structures



University of Colorado
Boulder

Reminders

Topics

- More on Pointers and References
- Returning Pointers and References
- Array Doubling

References

- A reference is like an alias
- Shares the same memory address as the original variable
- A reference as another name for the same variable
- Remember!
  - Pointers can point to NULL
  - Pointers can iterate over an array (++, --)
  - Pointers can be re-assigned
  - Must initialize reference when they are declared
  - Pointers are dereferenced whereas references aren’t

References

```
int main()
{
    int x = 25;
    int &q = x;

    q = 100;

    cout << &x << endl;
    cout << &q << endl;
    cout << "x: " << x << " q: " << q << endl;

    return 0;
}
```

Pass by Pointer (again)

```
void func(int *x)
{
    // *x = 3;
    x = 0;
}
```

```
int main()
{
    int a = 1;
    int *ptr = &a;

    func(ptr);

    cout << *ptr << endl;

    return 0;
}
```

Assigning Pointers

```
int main()
{
   int x=2, y=3;

   int *px=&x, *py=&y, *s;

   s=py;

   py=px;

   *px = *py+2;

   cout << x << " " << y << " " << *s << " " << *px << endl;

   return 0;
}
```

Pass Pointer by Reference

```
int global = 100;

// change ref to ptr
void func(int *& x)
{
    x = &global;
}
```

```
int main()
{
    int var = 3;
    int *ptr_to_var = &var;

    cout << "Before :" << *ptr_to_var << endl;

    func(ptr_to_var);

    cout << "After :" << *ptr_to_var << endl;

    return 0;
}
```

Output:

```
Before: 3
After: 100
```

Pass Pointer to Reference

- Note:
  - As we saw, passing a reference to a pointer is possible
    - *&
  - Passing a pointer to a reference **is not**!
    - &*
  - We can pass pointers to pointers! (similar to ref to ptr)
    - **

Returning Pointers

```
int* createArray() {
   int arr_ca[2];
   arr_ca[0] = 50;
   arr_ca[1] = 100;
   return arr_ca;
}

int main()
{
   int *arr = createArray();
   cout << arr[0] << endl;
   cout << arr[1] << endl;
}
```

Returning Pointers

```
int* createArray() {
    int* arr_ca = new int[2];
    arr_ca[0] = 50;
    arr_ca[1] = 100;
    return arr_ca;
}

int main()
{
    int *arr = createArray();
    cout << arr[0] << endl;
    cout << arr[1] << endl;
    …
    delete[] arr;
    arr = nullptr;
}
```

Returning References

```
int global = 99;

int& foo1()
{
    static int x = 5;
    return x;
}

int& foo2()
{
   return global;
}

int main()
{
   foo1() = 10;
   cout << foo1() << endl;
   cout << foo2() << endl;

   return 0;
}
```

Return Reference to Pointer

```
double *& getNum()
{
   double num = 1000.00;
   static double *x = &num;
   return x;
}

int main()
{
   double N = *getNum();
   cout << N << endl;       // 1000.00
}
```

Array Doubling

- Vectors can grow and shrink in size
- Not quite the same with arrays in C++

Questions