## CSI 33

In our work today we will review:

- assignment statements
- loops and control structures
- functions
- arrays in C++, lists in Python

## Functions in Python

- can return as many values as needed
- their parameters/arguments are passed by value only, however mutable objects can be modified inside a function
- parameters with default values must be at the end of the list of parameters

```
def it(myInt1, myFloat1, myList, myString)
```

## Functions in C++

- can return only one value
- have declaration and definition
- their parameters/arguments can be passed by value or by reference; in addition we can declare parameters as `const`

  not to allow modification inside a function
- arrays are automatically passed by reference (no & is used)
- parameters with default values must be at the end of the list of parameters

## Functions in C++

```
void f(const int a, const int &b, int &c, int d=3)
{
  a = 2; // will generate a compiler error
  b = 5; // will generate a compiler error
  c = 12; // the change affects actual parameter
  d = 13; // will not affect the actual parameter
}
```

## #1 Linear Search

- Let’s implement the following algorithm that performs *linear search* of a value `target` in the list/array/sequence of `items`.
- If the `target` is present, it returns its position, otherwise it returns -1:
- It should be defined as a function!

```
linearSearch(items, target):
   i = 0
   while i < len(items):
      if items[i] == target:  return i
      i += 1
   return -1
```

## #2 Binary Search

- Let’s use the Python implementation the *binary search* of a value `target` in the ordered sequence of `items`.
- If the `target` is present, it returns its position, and -1 otherwise.
- We need to write a C++ implementation, using C style arrays

```
def binarySearch(items, target):
  low = 0, high = len(items) - 1
  while low <= high:
     mid = $\left\lfloor \frac{low+high}{2} \right\rfloor$
     midItem = items[mid]
     if target == midItem:
        return mid
     else if target < item:
        high = mid - 1
     else:
        low = mid + 1
  return -1
```

## #3 Selection sort (page 37, exercise 3, Chapter 1)

- Let’s implement Selection sort in Python and in C++

The Selection Sort algorithm sorts a list by finding the smallest element and swapping it with the element in position zero of the list.

It then finds the next smallest element and swaps it with the element in position one of the list.

This process repeats until we have found the n - 1th smallest element and put it in position n - 2. At this point , the largest element is in position n - 1 .                    see the handout