Contents

1 Notebook.cpp

```
/*Basics*/

//Vectors are passed by copy in C++, unlike arrays which are passed by reference
//get in C++
//Returns the i-th element in a tuple
get<i>(array_element);

//reading chars
Recommend scanf("␣%c") to avoid newline. Otherwise newline is counted

//scanf("%s") will pick up strings(reading characters up until whitespace) then
    append a \0 at the end regardless of array length.


/*GCD query*/
//Just remember to please make a bigger than b
int gcd(int a, int b) {return (a%b)==0? b:gcd(b, a%b);}


/*qsort or mergesort*/
//Assuming we're sorting in ascending order from left to right, based on integer
values. First thing to do is define a comparator function, and then call it.
    Note that mergesort has the same format, except obviously called with
    mergesort
int comparator(const void * a, const void * b){

        int c = *(const int*)a;
        int d = *(const int*)b;

        if(c > d){
                return 1;
        } else if(c == d){
                return 0;
        } else{
                return -1;
        }
}

//...In the main function
qsort(array_name, size_of_array, size_of_array_elements, comparator);


/* Binary Search*/
//For if we just want to know if an value exists, if array is unsorted, if you
    want to return the index, then do so

int binary_search(int target, int * array){

        qsort(array);
        int found = 0;
        int lower, middle, upper;
        lower = 0; upper = (int)sizeof(array)-1; middle = (lower+upper)/2;
        while(1){
                if(array[middle] == target){
                        found = 1;
```

```
                        break;
                } else if (middle == lower){
                        if(array[upper] == target){
                                found = 1;
                        }
                        break;
                } else if (array[middle] < target) {
                        lower = middle; middle=(lower+upper)/2;
                } else{
                        upper = middle; middle = (lower+upper)/2;
                }
        }
        return found;
}


/*Dictionaries in Python*/

//Multidimensional
dict1["dict2"]["key"] = val //{dict2: {"key": val}}
```