■ C Programming & Algorithms Cheat Sheet

■ C Fundamentals

```
Structure of a C program:
#include <stdio.h>
int main() {
    printf("Hello, World!");
    return 0;
}

Data Types:
int, float, double, char, void

Operators:
Arithmetic: + - * / %
Relational: == != > < >= <=
Logical: && || !
Assignment: = += -= *= /= %=

Control Statements:
if (condition) { ... }
else if (...) { ... }
else { ... }

switch(x) {
    case 1: ...; break;
    default: ...;
}

for (int i=0; i<n; i++) { ... }
while (condition) { ... }
do { ... } while (condition);

Functions:
returnType functionName(params) { ... }

Arrays:
int arr[5] = {1,2,3,4,5};

Pointers:
int x = 10, *p = &x;
printf("%d", *p);  // dereference
```

■ Bubble Sort

```
Ascending:
for (int i=0; i<n-1; i++) {
    for (int j=0; j<n-i-1; j++) {
        if (arr[j] > arr[j+1]) {
            int temp = arr[j];
            arr[j] = arr[j+1];
```

```
            arr[j+1] = temp;
        }
    }
}
```

Descending → change > to <

■ Selection Sort

```
Ascending:
for (int i=0; i<n-1; i++) {
    int min = i;
    for (int j=i+1; j<n; j++) {
        if (arr[j] < arr[min])
            min = j;
    }
    int temp = arr[min];
    arr[min] = arr[i];
    arr[i] = temp;
}
```

Descending → change < to >

■ Insertion Sort

```
Ascending:
for (int i=1; i<n; i++) {
    int key = arr[i];
    int j = i-1;
    while (j >= 0 && arr[j] > key) {
        arr[j+1] = arr[j];
        j--;
    }
    arr[j+1] = key;
}
```

Descending → change > to <

■ Recursion

```
Factorial:
int fact(int n) {
    if (n == 0) return 1;
    return n * fact(n-1);
}

Fibonacci:
int fib(int n) {
    if (n <= 1) return n;
    return fib(n-1) + fib(n-2);
}

Key Rules:
```

1. Must have a base case.
2. Each call should move towards base case.
3. Can replace loops, but slower for large inputs.