1. Example Code

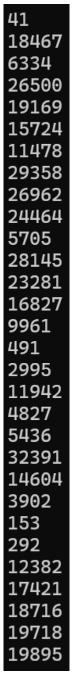
```
void problemOne() {
    srand(50);
    int nums[50];
    for (int i = 0; i < 50; i++) {
        nums[i] = 100 + (rand() % 101);
        std::cout << nums[i] << "    ";
    }
}
```

What its doing: Setting the seed for random to 50 then generating 50 random numbers between 100 to 200. Because the seed isn't changed between runs the numbers will be the same for each run.

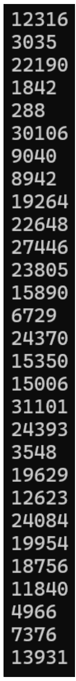
2. Learning about rand() and srand()

```
void problemTwo() {
    for (int i = 0; i < 30; i++) {
        std::cout << rand() << std::endl;
    }
}
```

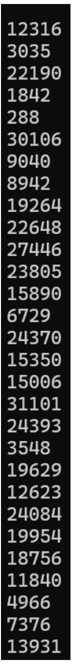
Output:



```
41
18467
6334
26500
19169
15724
11478
29358
26962
24464
5705
28145
23281
16827
9961
491
2995
11942
4827
5436
32391
14604
3902
153
292
12382
17421
18716
19718
19895
```



```
12316
3035
22190
1842
288
30106
9040
8942
19264
22648
27446
23805
15890
6729
24370
15350
15006
31101
24393
3548
19629
12623
24084
19954
18756
11840
4966
7376
13931
```

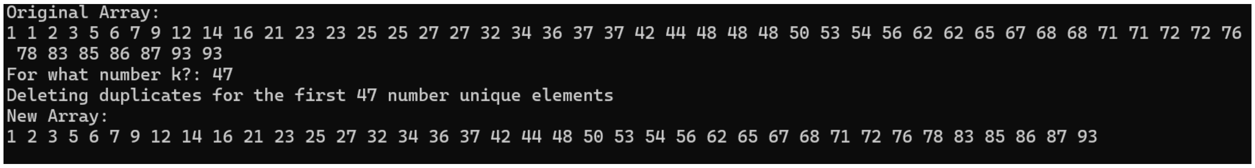


```
12316
3035
22190
1842
288
30106
9040
8942
19264
22648
27446
23805
15890
6729
24370
15350
15006
31101
24393
3548
19629
12623
24084
19954
18756
11840
4966
7376
13931
```

3. Pseudo-random integer numbers

```
void problemThree() {
    srand(3);
    int nums[50];
    for (int i = 0; i < 50; i++) {
        nums[i] = rand() % 101;
    }
    int size = sizeof(nums) / sizeof(int);

    // Sorting
    for (int i = 0; i < size - 1; ++i) {
        for (int j = 0; j < size - i - 1; ++j) {
            if (nums[j] > nums[j + 1]) {
                int temp = nums[j];
                nums[j] = nums[j + 1];
                nums[j + 1] = temp;
            }
        }
    }

    // Outputting Orginal Array
    std::cout << "Original Array:\n";
    for (int i = 0; i < size; i++) {
        std::cout << nums[i] << " ";
    }
    std::cout << std::endl;

    // Getting user input for k
    int k = 0;
    std::cout << "For what number k?: ";
    std::cin >> k;
    std::cout << "Deleting duplicates for the first " << k << " number unique 
elements\n";

    // Deleting duplicates up to k elements
    int newSize = size;
    for (int i = 0; i < k; ++i) {
        for (int j = i + 1; j < newSize; ++j) {
            if (nums[i] == nums[j]) {
                for (int z = j; z < newSize - 1; ++z) {
                    nums[z] = nums[z + 1];
                }
                --newSize;
                --j;
            }
        }
    }

    std::cout << "New Array:\n";
    for (int i = 0; i < newSize; i++) {
        std::cout << nums[i] << " ";
    }
    std::cout << std::endl;
}
```

Output:

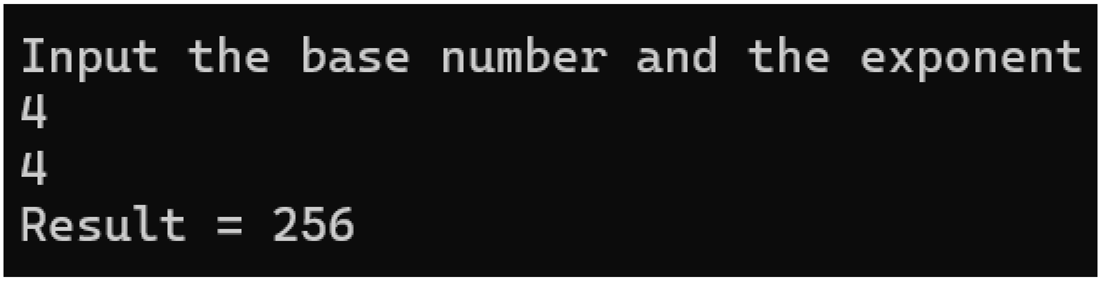


```
Original Array:
1 1 2 3 5 6 7 9 12 14 16 21 23 23 25 25 27 27 32 34 36 37 37 42 44 48 48 48 50 53 54 56 62 62 65 67 68 68 71 71 72 72 76
 78 83 85 86 87 93 93
For what number k?: 47
Deleting duplicates for the first 47 number unique elements
New Array:
1 2 3 5 6 7 9 12 14 16 21 23 25 27 32 34 36 37 42 44 48 50 53 54 56 62 65 67 68 71 72 76 78 83 85 86 87 93
```

4. Recursion 1

```
void problemFour() {
    int b, e;
    std::cout << "Input the base number and the exponent" << std::endl;
    std::cin >> b >> e;
    std::cout << "Result = " << power(b, e) <<std::endl;
}

int power(int b, int e) {
    if (e == 0) {
        return 1;
    }
    return b * power(b, e - 1);
}
```

Output:

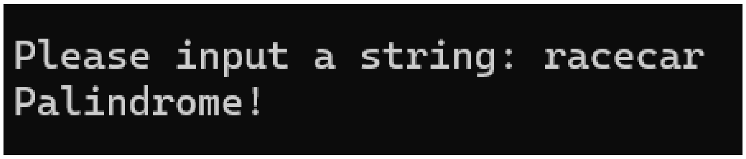


```
Input the base number and the exponent
4
4
Result = 256
```

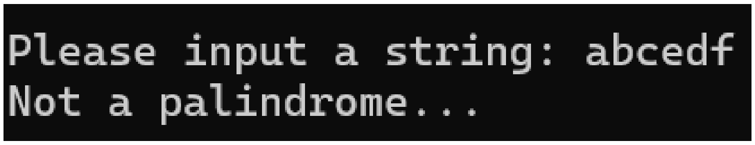
5. Recursion 2

```
void problemFive() {
    std::string input;

    std::cout << "Please input a string: ";
    std::cin >> input;



    std::cout << (palindrome(input,0,input.length()-1) ? "Palindrome!" : "Not a
palindrome...");
}

bool palindrome(std::string s, int start, int end) {
    if (start >= end) {
        return true;
    }
    else if (s[start] == s[end]) {
        palindrome(s, ++start, --end);
    }
    else
        return false;
}
```

Output:



```
Please input a string: racecar
Palindrome!
```

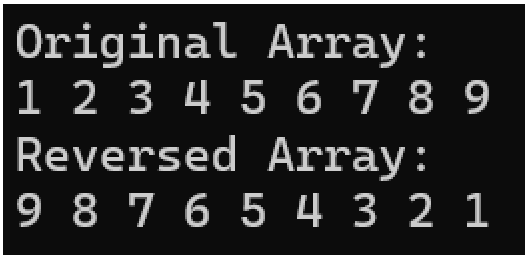


```
Please input a string: abcedf
Not a palindrome...
```

6. Reverse array elements

```
void problemSix(){
    int nums[] = { 1,2,3,4,5,6,7,8,9 };
    size_t size = sizeof(nums) / sizeof(int);

    std::cout << "Original Array:\n";
    for (int i = 0; i < size; i++) {
        std::cout << nums[i] << " ";
    }
    std::cout << std::endl;

    reverse_array(nums, size);

    std::cout << "Reversed Array:\n";
    for (int i = 0; i < size; i++) {
        std::cout << nums[i] << " ";
    }
    std::cout << std::endl;
}

void reverse_array(int array[], size_t arraySize) {
    int start = 0, end = arraySize - 1;
    int temp = 0;

    while (start < end) {
        temp = array[start];
        array[start] = array[end];
        array[end] = temp;
        start++;
        end--;
    }
}
```

Output:



```
Original Array:
1 2 3 4 5 6 7 8 9
Reversed Array:
9 8 7 6 5 4 3 2 1
```