



## Greedy Algorithms

### 1-G-Coin Problem:

Question 1 | Correct Mark 1.00 out of 1.00 [Flag question](#)

Write a program to take value  $V$  and we want to make change for  $V$  Rs, and we have infinite supply of each of the denominations in Indian currency, i.e., we have infinite supply of  $\{1, 2, 5, 10, 20, 50, 100, 500, 1000\}$  valued coins/notes, what is the minimum number of coins and/or notes needed to make the change.

Input Format:

Take an integer from stdin.

Output Format:

print the integer which is change of the number.

Example Input :

64

Output:

4

Explanaton:

We need a 50 Rs note and a 10 Rs note and two 2 rupee coins.

```

1  #include <stdio.h>
2
3  int main() {
4      int V;
5      scanf("%d", &V);
6
7      int m[] = {1000, 500, 100, 50, 20, 10, 5, 2, 1};
8      int n = sizeof(m) / sizeof(m[0]);
9      int count = 0;
10     for (int i = 0; i < n; i++) {
11         while (V >= m[i]) {
12             V -= m[i];
13             count++;
14         }
15     }
16     printf("%d\n", count);
17     return 0;
18 }
19

```

	Input	Expected	Got	
✓	49	5	5	✓

Passed all tests! ✓

2-G-Cookies Problem:

**Question 1** | Correct Mark 1.00 out of 1.00 [Flag question](#)

Assume you are an awesome parent and want to give your children some cookies. But, you should give each child at most one cookie.

Each child  $i$  has a greed factor  $g[i]$ , which is the minimum size of a cookie that the child will be content with; and each cookie  $j$  has a size  $s[j]$ . If  $s[j] \geq g[i]$ , we can assign the cookie  $j$  to the child  $i$ , and the child  $i$  will be content. Your goal is to maximize the number of your content children and output the maximum number.

**Example 1:**

**Input:**

3

1 2 3

2

1 1

**Output:**

1

Explanation: You have 3 children and 2 cookies. The greed factors of 3 children are 1, 2, 3.

And even though you have 2 cookies, since their size is both 1, you could only make the child whose greed factor is 1 content.

You need to output 1.

**Constraints:**

$1 \leq g.length \leq 3 \times 10^4$

$0 \leq s.length \leq 3 \times 10^4$

$1 \leq g[i], s[j] \leq 2^{31} - 1$

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int cmp(const void *a, const void *b) { return (*(int*)a - *(int*)b); }
5
6  int main() {
7      int n, m, i = 0, j = 0, ans = 0;
8      scanf("%d", &n);
9      int g[n]; for(int k=0;k<n;k++) scanf("%d",&g[k]);
10     scanf("%d", &m);
11     int s[m]; for(int k=0;k<m;k++) scanf("%d",&s[k]);
12
13     qsort(g, n, sizeof(int), cmp);
14     qsort(s, m, sizeof(int), cmp);
15
16     while(i < n && j < m){
17         if(s[j] >= g[i]){ ans++; i++; j++; }
18         else j++;
19     }
20     printf("%d\n", ans);
21     return 0;
22 }
```

	Input	Expected	Got	
✓	2	2	2	✓
	1 2			
	3			
	1 2 3			

Passed all tests! ✓

4-G-Array Sum max problem:

**Question 1** | Correct | Mark 1.00 out of 1.00 | [Flag question](#)

Given an array of N integer, we have to maximize the sum of  $arr[i] * i$ , where i is the index of the element ( $i = 0, 1, 2, \dots, N$ ). Write an algorithm based on Greedy technique with a Complexity  $O(n \log n)$ .

Input Format:

First line specifies the number of elements-n

The next n lines contain the array elements.

Output Format:

Maximum Array Sum to be printed.

Sample Input:

5

2 5 3 4 0

Sample output:

40

**Answer:** (penalty regime: 0 %)

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int compare(const void *a, const void *b) {
5     int x = *(int*)a;
6     int y = *(int*)b;
7     return x - y;
8 }
9
10 int main() {
11     int n;
12     scanf("%d", &n);
13     int *arr = (int*)malloc(n * sizeof(int));
14     for (int i = 0; i < n; i++) {
15         scanf("%d", &arr[i]);
16     }
17
18     qsort(arr, n, sizeof(int), compare);
19
20     long long sum = 0;
21     for (int i = 0; i < n; i++) {
22         sum += (long long)arr[i] * i;
23     }
24
25     printf("%lld\n", sum);
26
27     free(arr);
28     return 0;
29 }
```

	Input	Expected	Got	
✓	5 2 5 3 4 0	40	40	✓
✓	10 2 2 2 4 4 3 3 5 5 5	191	191	✓
✓	2 45 3	45	45	✓

Passed all tests! ✓

## 5-G-Product of Array elements-Minimum:

Question 1 | Correct Mark 1.00 out of 1.00 [Flag question](#)

Given two arrays `array_One[]` and `array_Two[]` of same size `N`. We need to first rearrange the arrays such that the sum of the product of pairs( 1 element from each) is minimum. That is  $\text{SUM}(A[i] * B[i])$  for all `i` is minimum.

**For example:**

Input	Result
3 1 2 3 4 5 6	28

```

1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int compareAsc(const void *a, const void *b) {
5      int x = *(int*)a;
6      int y = *(int*)b;
7      return x - y;
8  }
9
10 int compareDesc(const void *a, const void *b) {
11     int x = *(int*)a;
12     int y = *(int*)b;
13     return y - x;
14 }
15
16 int main() {
17     int n;
18     scanf("%d", &n);
19
20     int *A = (int*)malloc(n * sizeof(int));
21     int *B = (int*)malloc(n * sizeof(int));
22
23     for (int i = 0; i < n; i++) {
24         scanf("%d", &A[i]);
25     }
26     for (int i = 0; i < n; i++) {
27         scanf("%d", &B[i]);
28     }
29
30     qsort(A, n, sizeof(int), compareAsc);
31     qsort(B, n, sizeof(int), compareDesc);
32
33     long long sum = 0;
34     for (int i = 0; i < n; i++) {
35         sum += (long long)A[i] * B[i];
36     }
37
38     printf("%lld\n", sum);
39     free(A);
40     free(B);
41     return 0;
42 }
43

```

	Input	Expected	Got	
✓	3 1 2 3 4 5 6	28	28	✓
✓	4 7 5 1 2 1 3 4 1	22	22	✓
✓	5 20 10 30 10 40 8 9 4 3 10	590	590	✓

Passed all tests! ✓



