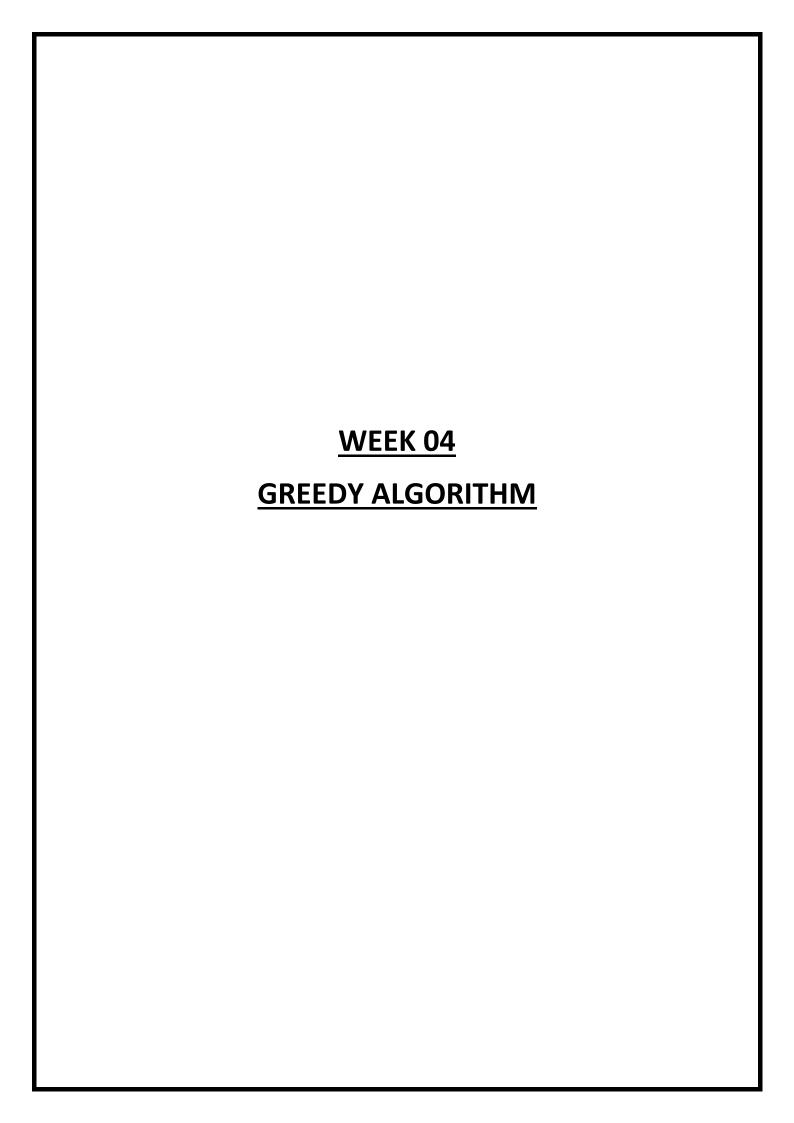
## RAJALAKSHMI ENGINEERING COLLEGE RAJALAKSHMI NAGAR, THANDALAM – 602 105



# CS23331 DESIGN AND ANALYSIS OF ALGORITHM LAB

## **Laboratory Observation Note Book**

Name : Amirtha Raja Rajeswari N.A.
Year / Branch / Section : . 2 <sup>nd</sup> Year / AIML / A
Register No. : 231501012
Semester : 3 <sup>rd</sup> Şemester
Academic Year : 2024-2025.



1) 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:** 

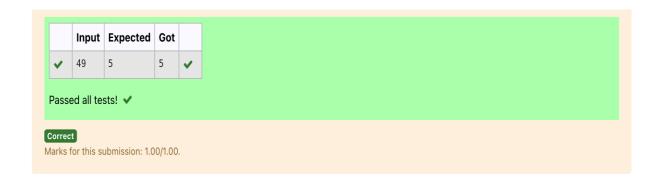
4

**Explanation:** 

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

```
#include <stdio.h>
int main()
{
   int cost;
   scanf("%d",&cost);
   int coin[9] = {1,2,5,10,20,50,100,500,1000};
   int i=0, count= 0;
```

```
for (i=9-1; i>0; i--) {
    while (cost >= coin[i]) {
       cost -= coin[i];
       count++;
    }
    printf("%d",count);
}
```



2) 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] >= 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

123

2

11

**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.

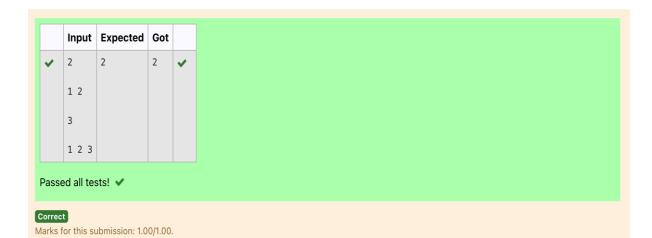
You need to output 1.

#### **Constraints:**

```
1 <= g.length <= 3 * 10^4
```

$$1 \le g[i], s[j] \le 2^31 - 1$$

```
#include<stdio.h>
int main(){
  int chno,cono;
  int satisfied=0,j=0;
  scanf("%d",&chno);
  int child[chno];
  for(int i = 0;i<chno;i++){</pre>
    scanf("%d",&child[i]);
  }
  scanf("%d",&cono);
  int cookie[cono];
  for(int i = 0; i<cono;i++){</pre>
    scanf("%d",&cookie[i]);
  for(int i=0;i<chno;i++){</pre>
    if(child[i]<=cookie[j]){</pre>
       satisfied+=1;
       j++;
    }
  printf("%d",satisfied);
}
```



3) A person needs to eat burgers. Each burger contains a count of calorie. After eating the burger, the person needs to run a distance to burn out his calories.

If he has eaten i burgers with c calories each, then he has to run at least  $3^i * c$  kilometers to burn out the calories. For example, if he ate 3

burgers with the count of calorie in the order: [1, 3, 2], the kilometers he needs to run are  $(3^0 * 1) + (3^1 * 3) + (3^2 * 2) = 1 + 9 + 18 = 28$ .

But this is not the minimum, so need to try out other orders of consumption and choose the minimum value. Determine the minimum distance

he needs to run. Note: He can eat burger in any order and use an efficient sorting algorithm. Apply greedy approach to solve the problem.

**Input Format** 

First Line contains the number of burgers
Second line contains calories of each burger which is n spaceseparate integers

**Output Format** 

Print: Minimum number of kilometers needed to run to burn out the calories

**Sample Input** 

3 5 10 7

**Sample Output** 

76

## For example:

Test	Input	Result
Test Case 1	3	18
	132	

```
#include<stdio.h>
#include<math.h>
int main()
{
  int burg,i,j;
  scanf("%d",&burg);
  int cal[burg];
  for(i=0;i<burg;i++){</pre>
    scanf("%d",&cal[i]);
  }
  int temp,kms=0;
  for(i=0;i<burg-1;i++){
    for(j=0;j<burg-i-1;j++){
       if(cal[j]>cal[j+1]){
         temp=cal[j];
         cal[j]=cal[j+1];
```

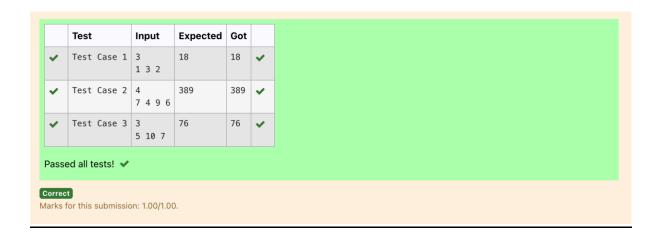
```
cal[j+1]=temp;
}

}

j=burg;

for(i=0;i<burg;i++){
    kms+=(pow(burg,i)*cal[j-1]);
    j--;
}

printf("%d",kms);
return 0;
}</pre>
```



4) 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, ..., N). Write an algorithm based on Greedy technique with a Complexity O(nlogn).

#### **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

25340

Sample output:

40

```
#include<stdio.h>
int main()
{
    int N,temp,i,sum=0;
    scanf("%d",&N);
    int arr[N];
    for(i=0;i<N;i++){
        scanf("%d",&arr[i]);
    }
    for (int i = 0; i < N - 1; i++) {</pre>
```

```
int mind = i;
    for (int j = i + 1; j < N; j++) {
       if (arr[j] < arr[mind]) {</pre>
         mind = j;
       }
    }
    temp = arr[mind];
    arr[mind] = arr[i];
    arr[i] = temp;
  }
  for(i=0;i<N;i++){
    sum=sum+arr[i]*i;
  }
  printf("%d",sum);
  return 0;
}
```

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

Marks for this submission: 1.00/1.00.

5) 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 SUM (A[i] \* B[i]) for all i is minimum.

#### For example:

Input	Result
3	28
1	
2	
3	
4	
5	
6	

```
#include<stdio.h>
int main()
{
   int N,i,j,temp,sum,flag=0;
   scanf("%d",&N);
   int arr1[N];
   int arr2[N];
   for(i=0;i<N;i++){</pre>
```

```
scanf("%d",&arr1[i]);
}
for(i=0;i<N;i++){
  scanf("%d",&arr2[i]);
}
for (i=0;i<N-1;i++){
  for(j =0;j<N-i-1;j++){
    if(arr1[j]>arr1[j+1]){
       temp=arr1[j];
       arr1[j]=arr1[j+1];
       arr1[j+1]=temp;
    }
    if(arr2[j]>arr2[j+1]){
       temp=arr2[j];
       arr2[j]=arr2[j+1];
       arr2[j + 1] = temp;
    }
}
i=0;
j=N-1;
while(flag<N){
  sum+=arr1[i]*arr2[j];
  i++;
```

```
j--;
    flag++;
}
printf("%d",sum);
}
```

