**NAME:** SREYA G **REGISTER NO.:** 230701334

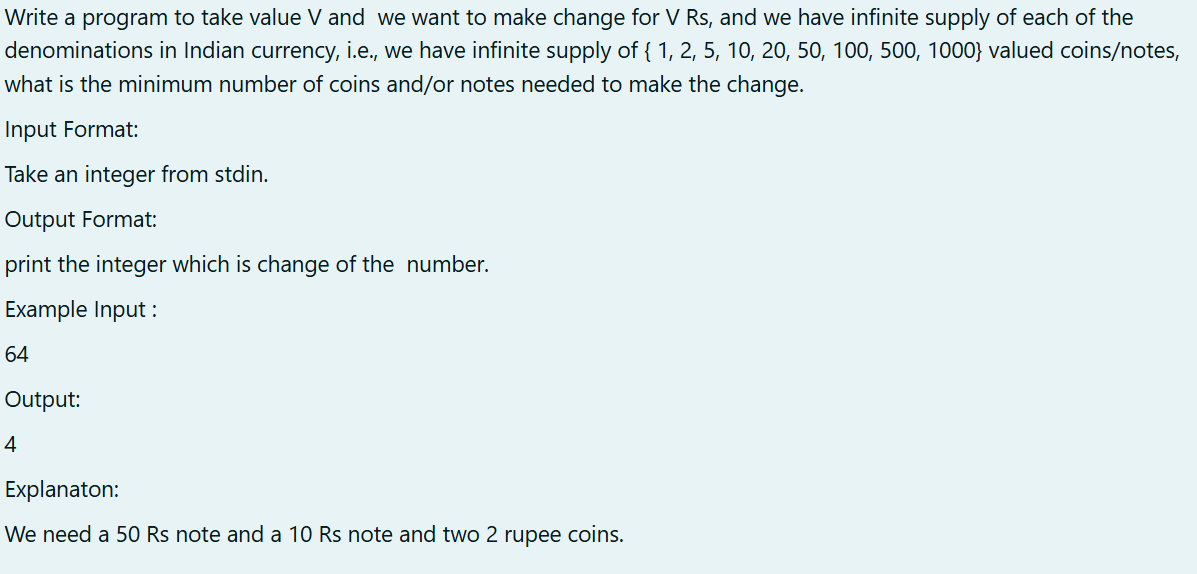
**CLASS :** CSE F **DATE:** 22/08/2024

**EX – 3:**

**GREEDY ALGORITHMS:**

PROBLEM 1:

AIM:



ALGORITHM:

1. Input V: Read the value .

2. Initialize denominations: Use an array of currency denominations in descending order.

3. Initialize count : Set count to zero.

4. Iterate through denominations:

- For each denomination:

- Add V / denomination to count.

- Update V to V % denomination.

5. Output count : Print the total count of notes/coins.

6. End.

CODE:

#include<stdio.h>

int main()

{

int V;

scanf("%d",&V);

int denominations[] = {1000,500,100,50,20,10,5,2,1};

int count = 0;

for(int i = 0;i < sizeof(denominations) / sizeof(denominations[0]);i++)

{

count += V/denominations[i];

V %= denominations[i];

}

printf("%d\n",count);

return 0;

}

OUTPUT:

A screenshot of a test

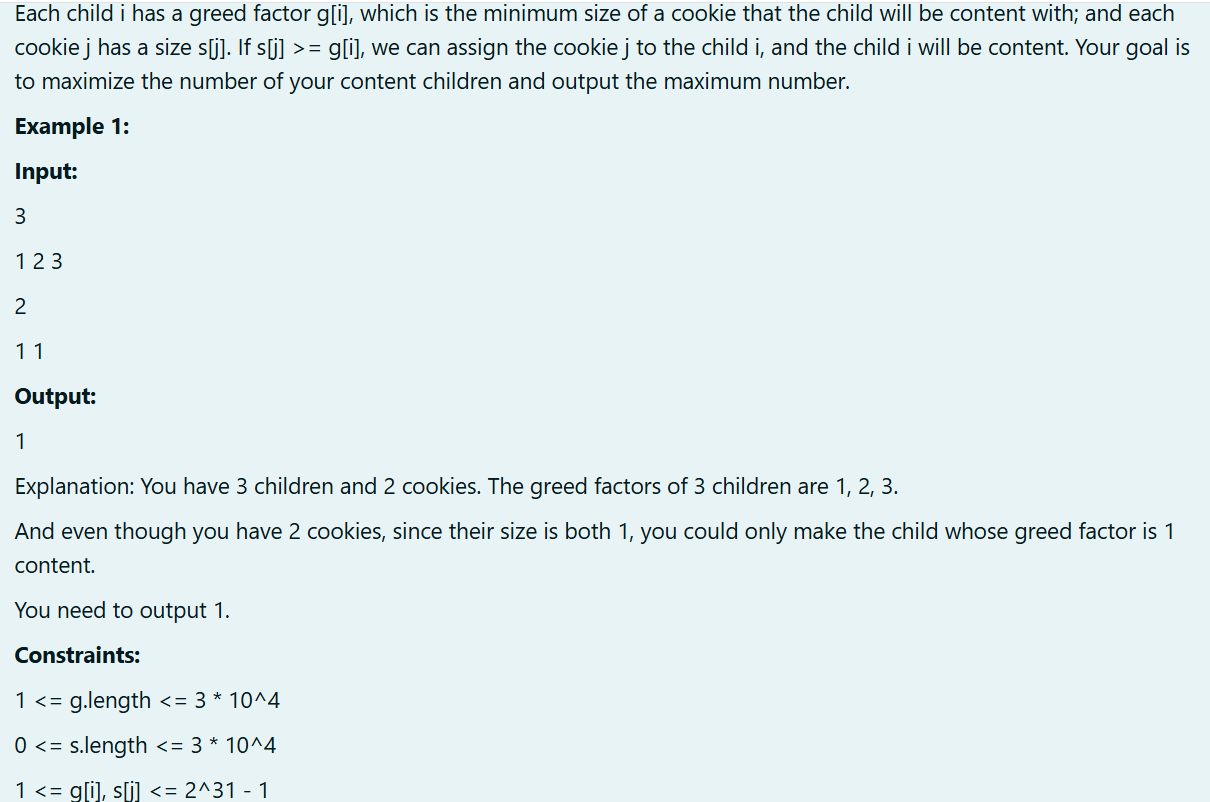
Description automatically generated

RESULT:

Thus the code is executed successfully and gives the expected output.

PROBLEM 2:

AIM:



ALGORITHM:

1. Input n and array g.

2. Input m and array c.

3. Initialize co = 0.

4. For each c[i], check if c[i] <= g[j] for any g[j]. If true, increment co and break.

5. Output co.

6. End.

CODE:

#include<stdio.h>

int main()

{

int n,m;

int co = 0;

scanf("%d",&n);

int g[n];

for(int i = 0;i < n;i++)

{

scanf("%d",&g[i]);

}

scanf("%d",&m);

int c[m];

for(int i = 0;i < m;i++)

{

scanf("%d",&c[i]);

}

for(int i = 0;i < n;i++)

{

for(int j = 0;j < m;j++)

{

if(c[i] <= g[j])

{

co++;

break;

}

}

}

printf("%d\n",co);

}

OUTPUT:

A screenshot of a test

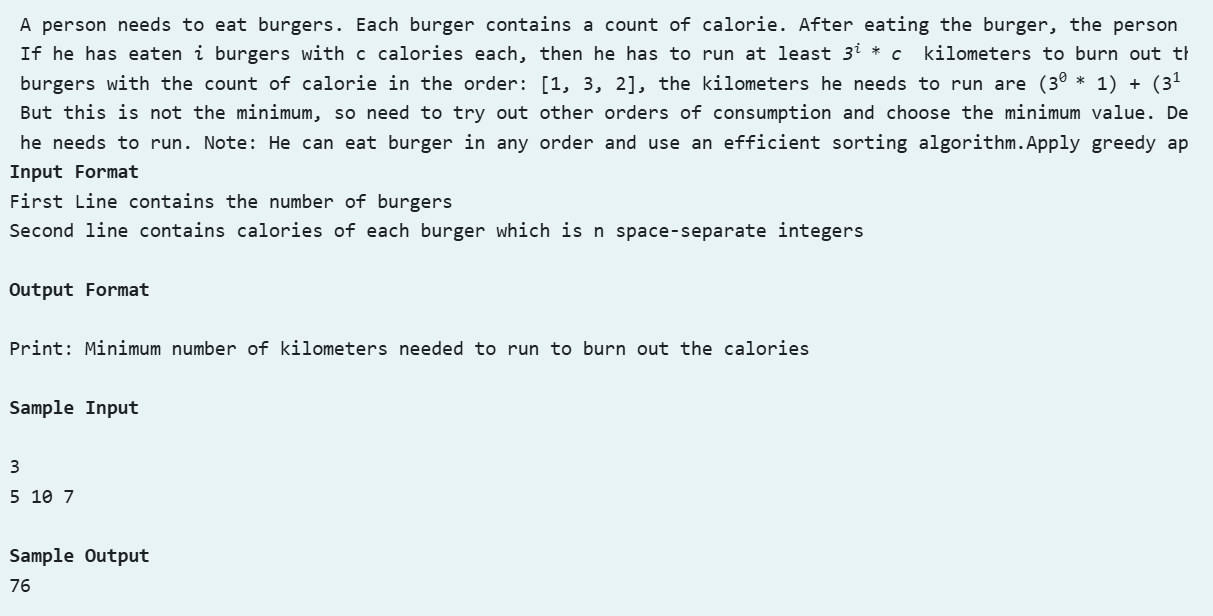
Description automatically generated

RESULT:

Thus the code is executed successfully and gives the expected output.

PROBLEM 3:

AIM:



A screenshot of a computer

Description automatically generated

ALGORITHM:

1. Input n and array a.

2. Sort a in descending order using Bubble Sort.

3. Initialize km = 0.

4. Calculate km:

- For each element a[i], add a[i] \* (n^i) to km.

5. Output km.

6. End.

CODE:

#include<stdio.h>

int main()

{

int n;

scanf("%d",&n);

int a[n];

for(int i = 0;i < n;i++)

{

scanf("%d",&a[i]);

}

int km = 0;

for(int i = 0;i < n-1;i++)

{

for(int j = 0;j < n-i-1;j++)

{

if(a[j] < a[j+1])

{

int t = a[j];

a[j] = a[j+1];

a[j+1] = t;

}

}

}

for(int i = 0;i < n;i++)

{

int p = 1;

if(i == 0)

km += (p\*a[0]);

else

{

for(int j = 1;j <= i;j++)

{

p \*= n;

}

km += (p \* a[i]);

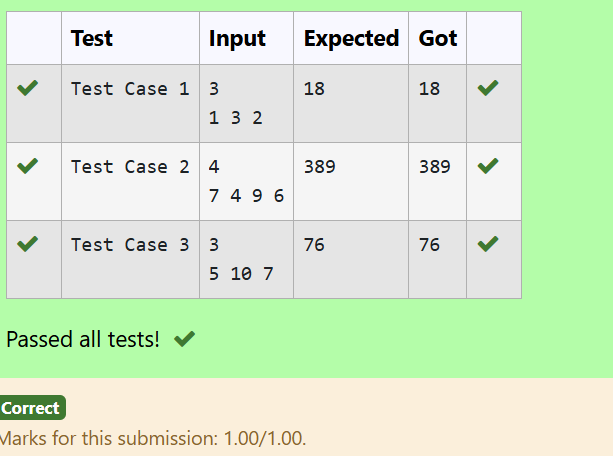
}

}

printf("%d",km);

}

OUTPUT:



RESULT:

Thus the code is executed successfully and gives the expected output.

PROBLEM 4:

AIM;

A screenshot of a computer

Description automatically generated

ALGORITHM:

1. Input n and array arr.

2. Sort arr in ascending order.

3. Initialize sum = 0.

4. Compute sum += arr[i] \* i for each element.

5. Output sum.

6. End.

CODE:

#include<stdio.h>

int main()

{

int n;

scanf("%d",&n);

int arr[n];

for(int i = 0;i < n;i++)

{

scanf("%d",&arr[i]);

}

for(int i = 0;i < n;++i)

{

for(int j = i+1;j < n;++j)

{

if(arr[i] > arr[j])

{

int a = arr[i];

arr[i] = arr[j];

arr[j] = a;

}

}

}

int sum = 0;

for(int i = 0;i < n;i++)

{

sum += arr[i]\*i;

}

printf("%d",sum);

}

OUTPUT:

A screenshot of a computer

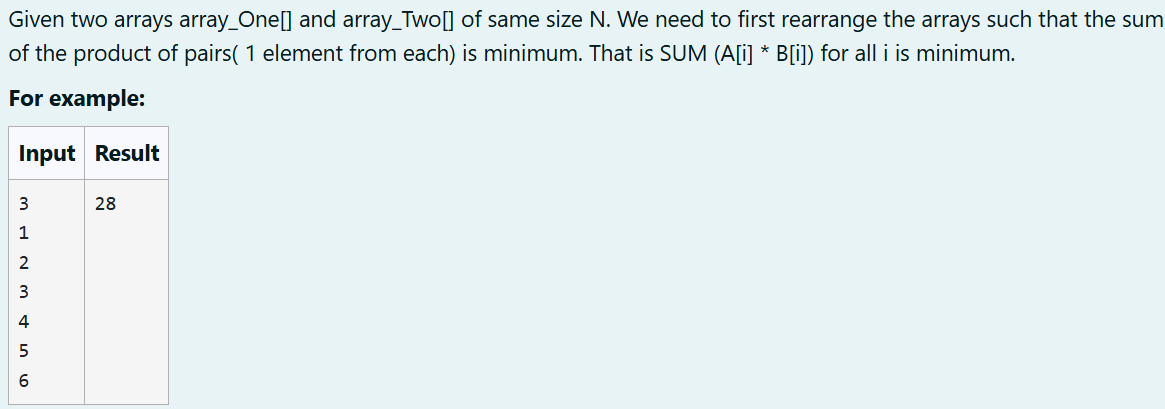
Description automatically generated

RESULT:

Thus the code is executed successfully and gives the expected output.

PROBLEM 5:

AIM:



ALGORITHM:

1. Input N and arrays A and B.

2. Sort A in ascending order using Bubble Sort.

3. Sort B in descending order using Bubble Sort.

4. Initialize s = 0.

5. Compute the sum: Add A[i] \* B[i] to s for all elements.

6. Output s.

7. End.

CODE:

#include<stdio.h>

int main()

{

int N;

scanf("%d",&N);

int A[N],B[N];

for(int i = 0;i < N;i++)

{

scanf("%d",&A[i]);

}

for(int i = 0;i < N;i++)

{

scanf("%d",&B[i]);

}

for(int i = 0;i < N-1;i++)

{

for(int j = 0;j < N-i-1;j++)

{

if(A[j] > A[j+1])

{

int t = A[j];

A[j] = A[j+1];

A[j+1] = t;

}

if(B[j] < B[j+1])

{

int t = B[j];

B[j] = B[j+1];

B[j+1] = t;

}

}

}

int s = 0;

for(int i = 0;i < N;i++)

{

s += (A[i] \* B[i]);

}

printf("%d",s);

}

OUTPUT:

A screenshot of a computer

Description automatically generated

RESULT:

Thus the code is executed successfully and gives the expected output.