GREEDY ALGORITHMS

PROBLEM 3:

3-G-BURGER PROBLEM

AIM:

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.

CODE:

```
#include <stdio.h>

#include <math.h>

int main() {

   int a, x, y, t, sum = 0;

   scanf("%d", &a);

   int arr[a];

   for(int i = 0; i < a; i++) {

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

   }

   for(x = 0; x < a - 1; x++) {

      for(y = x + 1; y < a; y++) {

         if(arr[x] < arr[y]) {

            t = arr[x];

            arr[x] = arr[y];
```

```
            arr[y] = t;
          }
        }
    }
    for(int b = 0; b < a; b++) {
        t = (int)pow(3, b);
        sum += t * arr[b];
    }

    // Print the result
    printf("%d\n", sum);

    return 0;
}
```

INPUT:

TEST CASE 1

3

1 3 2

TEST CASE 2

3

5 10 7

OUTPUT:

18

76