additional prac

1


NYU EATS
The Weekly EATS
Late Night Breakfast
To wish you well on finals, NYU Eats will be hosting a **complimentary breakfast for meal plan holders TONIGHT, May 10th** from **10 pm - 12 am** at **NYU Eats at Downstein** and **Palladium**. Kick start your study session with us!
NYU Eats at Downstein Menu
Scrambled Eggs
Vegetable Egg Whites

```
1
#include <bits/stdc++.h>
using namespace std;

void print(vector<int> a)
{
    for(int i = 0; i < a.size(); ++i)
        printf("%d ", a[i]);
    printf("\n");
}

int cmp_dec(int i, int j)
{/* sort in decreasing order */
    return i > j;
}

int max_ball(vector<int> diameter, vector<int> length)
{
    sort(diameter.begin(), diameter.end(), cmp_dec); print(diameter);
    sort(length.begin(), length.end(), cmp_dec); print(length);
    int i = 0, j = 0, count = 0;
    while(i < diameter.size())
    {
        if(diameter[i] <= length[j])
        {
            ++count; ++i; ++j;
        }
        else
            ++i;
    }
```

```
    return count;
}

int main()
{
    vector<int> diameter = {4, 5, 5, 9, 1, 10, 2, 7};
    vector<int> length = {9, 9, 10, 5, 3, 1, 1, 1, 2};
    printf("max_ball = %d\n", max_ball(diameter, length));
}
```

```
2
graham(set q of points):
vector<point> p
p[0] = point with min y-coordinate
p[1], p[2], ..., p[m] are remaining points
sorted by polar angle in counterclockwise order around p[0]
stack<point> s
s.push(p[0])
s.push(p[1])
s.push(p[2])
for i = 3 to m
      while angle formed by points
      s.next_to_top(), s.top(), p[i] /* next_to_top() returns the point one entry below the top of stack s */
      make a nonleft (counterclockwise) turn
            s.pop()
            s.push(p[i])
return s
```

operations in graham:

1. sort n - 1 points
2. add each point once to stack
3. remove points at most once from stack

max if

1. input points are given in reversely sorted order
2. n operations cannot be increased or decreased
3. # removals becomes maximized when almost all

points are removed, need at least 3 points to remain

in stack since at least 3 points on boundary of convex hull

worst case for graham:

set of n points whose convex hull consists of 3 points

3

```
/* split all edges of weight 2 into two edges of weight 1 each */
new_graph(G)
      G' = (V', E')
      for vertex u in V(G)
            u' = u
            V'.add(u')
      for edge e(u, v) in E(G)
            if weight(e) == 2
                  V'.add(vertex uv)
                  E'.add(e'(u', uv))
                  E'.add(e'(uv, v'))
            else
```

```
                E'.add(e'(u', v'))
      return G'

G' = new_graph(G)
bfs(G', src vertex s')
```

4

```
/* prim tc = O(E lg V) since
|V| * extract_min() = V lg V
|E| * decrease_key() = E lg V */
instead of min priority queue q using heap
consider set of edges whose weight is 1, 10, then 25
in extract_min(), extract element from 1-set, 10-set, 25-set take O(1) each time
O(|V| + |E|)

mst_prim(G, source vertex s)
      s.key = 0
      min heap queue Q := G.V
      vector<edge> T

      while Q is not empty
            r = Q.extract_min()
            T.push_back(edge (v, v.parent))
                  for each vertex u in adj[v] /* O(|E|) */
                        if u is in Q & u.key > weight(u, v)
                              u.key = weight(u, v)
                              u.parent = v
```

```
return T
```