Assignment #2

1.1) For $i \leftarrow 0$ to $n-2$ do

    $min \leftarrow i$

    for $j \leftarrow i+1$ to $n-1$ do

        if $A[j] < A[min]$ $min \leftarrow j$

        swap $A[i]$ and $A[min]$

$$\sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} 1 = \sum_{i=0}^{n-2} [(n-1)-(i+1)+i]$$

$$= \sum_{i=0}^{n-2} (n-1-i)$$

$$= \sum_{i=0}^{n-2} (n-1) - \sum_{i=0}^{n-2} i = (n-1)\sum_{i=0}^{n-2} 1 - \sum_{i=0}^{n-2} i$$

$$= (n-1)(n-1) + \frac{(n-2)(n-1)}{2} = (n-1)\left[(n-1) + \frac{(n-2)}{2}\right]$$

$$= (n-1)\left(\frac{n}{2}\right) = \frac{n(n-1)}{2} = \boxed{\frac{n^2}{2} - \frac{n}{2} \in \Theta(n^2)}$$

MERGE-(A,l,r)

1,2) if l<r

$m = \lfloor (q+r)/2 \rfloor$

MERGE-SORT(A, L, m)

MERGE-SORT(A, m+1, r)

MERGE(A,p,q,r)

$T(n) = 2T(n/2) + \Theta(n)$ $n = 2^k$

$a=2$ $b=2$ $d=1$

master

$2 = 2^1$

$a = b^d$

$T(n) \in \Theta(n^d \log n)$

$\boxed{T(n) \in \Theta(n \log n)}$

Bruteforce (Arr, n)

2.1)

```
for i ← 0   i ← n do
    for j ← i+1   j ← n do
        crossproduct();
        for k ← 0   k ← n do
                find if an extreme point
```

because it ask us to analyse it efficeency instead of solving, if that's what I'll do. The efficiency is (n^3) because for $n(n-1)/2$ point we have to analyse $n-2$ points because we don't need to analyse the points itself.

Shortest Path

```
for k ← 0 to 2
    d ← ∞
      for i ← i to n-1
          for j ← 0 to n do
              mind ← sqrt((xi-xj)^2 + (yi-yj)^2)
                checkval = a.x + b.y - c
              if (k==0)
                if (mind<d && checkval<0)
                    d ← mind
                      index=j
              else k==1
                if (mind<d && checkval<0)
                    d=mind
                    index=j
```

add point to array

$$C(n) = 2 \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} 1$$

$$= 2[(n-1) + (n-2) + \dots + 1] = 2(n-1)n \in \Theta(n^2)$$

2.2) quickHull (Arr, n, p1, p2)

for i=0 to n
findSides()
determine index
if not point then
return

FindHull (Arr, n, Arr[index], p1);
FindHull (Arr, n, Arr[index], p2);

$T(n) = 2T(n/2) + O(n)$

$a=2 \quad b=2 \quad d=1$

master

$2=2^1$

$a=b^d$

$T(n) \in \Theta(n^d \log n)$

$T(n) \in \Theta(n \log n)$

quickHull is $n \log n$ whereas brute force is n^3 making it so you always using quickhull since it's so much faster and take about the same time to implement