*1

These pages were added in case I didn't scan all of my hw pages. These are scratch work / rough drafts

~~DON'T SCAN~~

compare if

for j from 1 to n

if

$= V_1 + \max [M_4, M_6, M_7, M_8]$
$= V_2 + \max [M_6, M_7, M_8]$
$= V_3 + \max [M_7, M_8]$
$= V_4 + \max [M_8]$
$= V_5 + \max [M_8]$
$M_6) = V_6 + \max [\varnothing]$
$M_7) = V_7 + \max [\varnothing]$
$M_8) = V_8 + \max [\varnothing]$

Generate Lists

Set max value

total value

Set V_n as sum for a number i

*2

HW3 Rough Draft

job sorted by finish time

```
get_compatible_jobs( jobs_arr)
    for i from 1 to n
        for j from i+1 to n
            if jobs_arr[i].f <= jobs_arr[j].s
                jobs_arr[i].arr.append[j]
    get_max_value(jobs_arr[i])

    return max(jobs_arr[1].v, ..., jobs_arr[n].v)

get_max_value(job)
    if length(job.arr) = 0
        return
    else if length(job.arr) = 1
        job.v = job.v + job.arr[1].v; return
    else
        for i from 1 to length(job.arr)
            get_max_value(job.arr[i])
            return max of job.arr v
        job.v = job.v + max(job.arr[1].v, ... job.arr[le
        return
```

✳3

Case 2)

$$x_i = \begin{cases} 1 & i = 1, \dots, j-1 \\ x_j & i = j,\ x_j \in [0,1) \\ 0 & \text{otherwise} \end{cases}$$

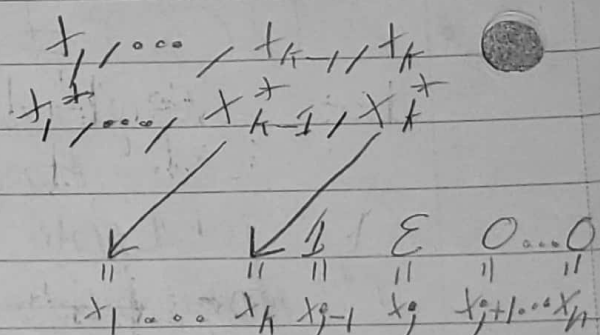
$$\sum_{i=1}^{j-1} w_i + \varepsilon w_j = W$$

- k is least index s.t. $x_k^* \neq x_k$

- $k \leq j$, $x_k = 1$

- $k = j$, $x_k^* = x_k$

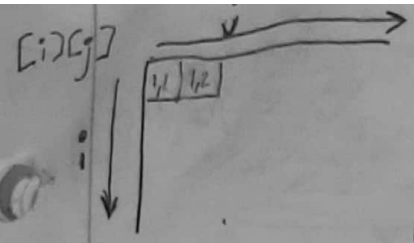
- $k > j$, $x_k^* = 0 = x_k$



$$x_k^* < x_j,\ x_k^* < 1$$

— increase x_k^* to x_k, decrease $x_{k+1}^*, \dots, x_n^*$ as needed. New solution $\{y_1, \dots, y_n\}$, and

$$\sum_{i \in [n]} \underbrace{w_i (x_i^* - y_i)}_{\text{weight decrease to } k\text{-}1\text{-}n \text{ item}} = \underbrace{w_k (y_k - x_k^*)}_{\text{weight increase to } k \text{ item}}$$



*4

CS 2500 HW3

3)

```
def find_longest_subsequence(str, n):
    max_palindrome[][] = arr(len=n, len=n)
    for i from 1 to n
        for j from 1 to n
            palindrome[r][c] = 0
        palindrome[r][1] = 1

    for sstr from 2 to n
        for r from 1 to n-sstr
            c = r+sstr-1
            if str[i] = str[j]
                if sstr = 2
                    max_palindrome[r][c] = 2
                else
                    max_palindrome[r][c] = max_palindrome[r][c] + 2
            else
                max_palindrome[r][c] = max(max_palindrome[r][c], max_palindrome[r+1][c])

    return max_palindrome[1][n]
```

*5

HW3 Rough Draft

4)

~~def distance(p1, p2)~~
~~return (p1.x - p2.x)^2 + (p1.y - p2.y)^2~~

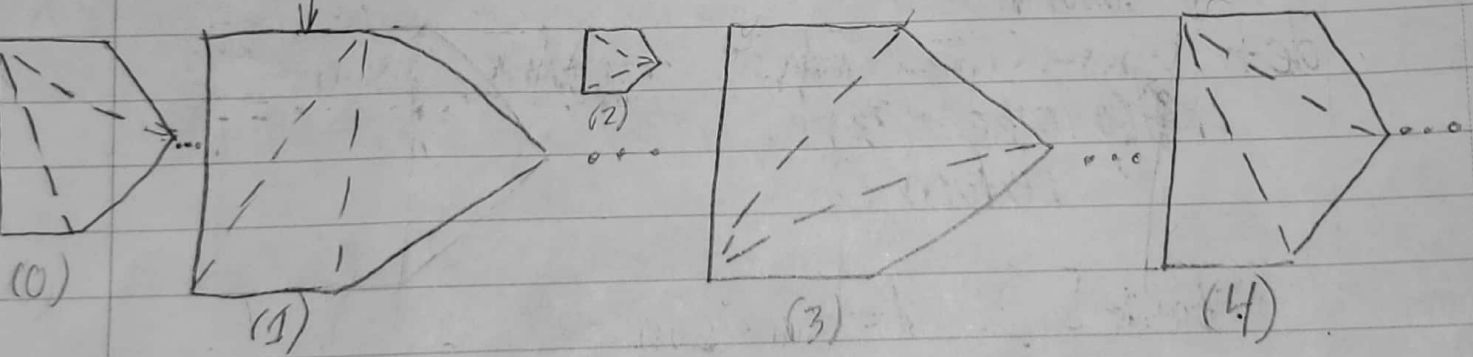
~~def minimum(x, y~~

~~2+1+√2+√2+1+√5+√5 = 4+2√2+2√5~~

+1 +2 +3

(2+1+√5) + (1+2+√5) + (2+√2+√2)

= 8 + 2√5 + 2√2



get perimeter of each triangle for a given triangulation

outer for loop ~~checks if points are called~~ goes through

*7

a → b

delete $s_i = D_i$
insert into $s_i = I_i$
replace s_i with $s_j = C_{ij}$

(2)

```
def dynamic_edit_distance(str1, l1, str2, l2, D, I, C):
    min_table[ ][ ] = arr[l1, l2]

    for r = 1 to l1+1
        for c = 1 to l2+1
        if r == 1 // if str1 is empty
            min_table[r, c] = sum(I[1 to c])
        else if c == 1 // if str2 is empty
            min_table[r, c] = sum(D[1 to r])
        else if str1[r-1] = str2[c-1]
            min_table[r, c] = min_table[r-1, c-1] + C[r-1] // = 0
                                insert str2[c-1] into str1[r]
        else
            min_table[r, c] = min(min_table[r][c-1] + I[c-1]
                                  min_table[r-1][c] + D[r-1] // delete str1[r-1]
                                  min_table[r-1][c-1] + C[r-1] // substitute str[r-1] w/ str[c-1]

    return min_table[r, c]
```

HW3 Rough Draft

(3)

```
def Greedy_Edit_Distance(str1, str2, l1, l2, D, I, C)
  min_table[ ][ ] = arr[l1+1, l2+1]

for r from 1 to l1+1
  for c from 1 to l2+1
    if r == 1
      min_table[r, c] = sum(I[1 to c])
    else if c == 1
      min_table[r, c] = sum(D[1 to r])
    else if str1[r-1] = str2[c-1]
      min_table[r, c] = C[r]
    else // different characters
      min_table[r, c] = min_table[r, c] + min(D[r-1], I[r-1], C[r-1])
                                        + min_table[r-1, c-1]

return min_table[r, c]
```

!?!?!

WRONG!

* 9

HW3 Rough Draft

(3)

```
def greedy_edit_distance(str1,str2,l1,l2,D,I,C)
 if l1 == 0
   return sum(I)
 else if l2 == 0
   return sum(D)
 else
    total_cost = 0

    cur_str1_len = l1
    for i=1 to l2
      if str1[i] == str2[i]
          tal_cost = total_cost
      else
        total_cost += min(D[i],C[i][
      if cur_str1_length < i
        total_cost += I[i]
        cur_str1_len += 1
      else
        if str1[i] == str2[i]
          total_cost += C[i]
        else
          min_val += min(C[i],D[i],I[i])
          total_cost += min_val
```

* 10

```
if C[i] != min_val
    if D[i] == min_val
        cur_str1_len -= 1
    else
        cur_str1_len += 1

for i from cur_str1_len to l2
    total_cost += D[i]

return total_cost
```