Graphs, Graph Algorithms, and Optimization
Problem set 4

W 24/25

GitHub https://github.com/JanPastorek/1-AIN-413-22-Graphs

Problem 0. [Any questions?]

Is there anything unclear from the lectures or about the home project?

Problem 1. [Huffman coding]

```
def huffman(ret):
    # frequency table
    ft = {}
    for c in ret:
        ft[c] = ft.get(c, 0) + 1

    # Priority queue, keys are freqs. of chars, values binary tries with freqs. of chars
    q = PriorityQueue()
    for c in set(ret):
        t = BinTree(c)
        q.add(ft[c], t)
    while len(q) > 1:
        f1, t1 = q.remove_min()
        f2, t2 = q.remove_min()
        t = BinTree(f1+f2, t1.root, t2.root)
        q.add(f1+f2, t)
    f, t = q.remove_min()
    return t
```

a) Compress these strings using primitive combinatorial coding and Huffman coding:

“abracadabra”

“anakonda”

“AATCGAACCTAGGA”

“TTAGCTTGGATCCT”

b) Decode these strings using the last two Huffman codings from a) (4 cases):

“AATAACCGGATTACCTAGGA”

“TTATTGGCCTAATGGATCCT”

Problem 2. [Binary trees]

Draw binary trees that represent these algebraic expressions:

a) $(1 + x)(1 - xy)$

b) $ln\left(\frac{1+x}{1-x}\right)$

Problem 3. [Merge sort]

```
def merge_sort(array):
    if len(array) < 2:
        return
    stred = len(array)//2
    array1 = array[:stred]
    array2 = array[stred:]
    merge_sort(array1)
    merge_sort(array2)
    # merging part
    i = j = 0
    while i + j < len(array):
        if j == len(array2)
        or i < len(array1)
        and array1[i] < array2[j]:
            array[i+j] = array1[i]
            i += 1
        else:
            array[i+j] = array2[j]
            j += 1
```

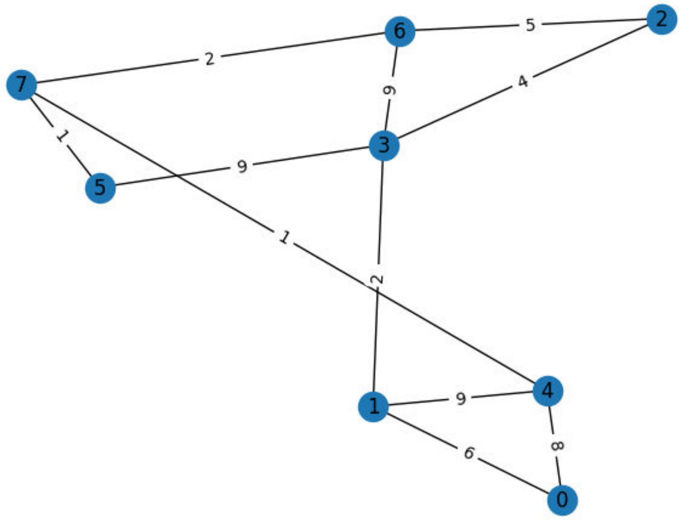
a) Sort the following arrays with the merge_sort.

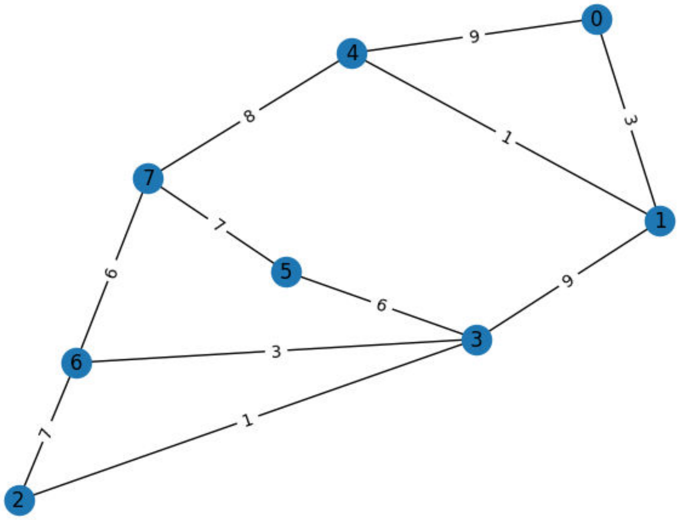
- [1,9,2,4,3,3,8,10,5]
- [13,0,1,5,2,3,2,4]

Problem 4. [Dijkstra - distances]

```
def dijkstra(v1):
    # dict D stores distances from v1 to other vertices
    D[all_vertices] = inf
    D[v1] = 0
    queue = priority queue of vertices based on D
    while queue:
        v1 = queue.remove_min()
        for v2 in neighbours(v1):
            if v2 is in queue:  # not visited yet
                if D[v1] + weight(v1, v2) < D[v2]:
                    D[v2] = D[v1] + weight(v1, v2)
    return D
```

a) Try dijkstra on various vertices of the following graphs


6
5
2
7
2
9
4
1
3
9
5
1
2
4
1
9
6
8
0


0
4
9
8
3
1
7
1
7
6
5
6
9
3
6
3
7
1
2