Greedy activity pseudocode

GREEDY-ACTIVITY-SELECTOR(s, f)

1 $n = s.length$ // n is the total number of activities
2 $A = \{a_1\}$ // Selection of first activity
3 $k = 1$ // current activity denoted by k and we consider it $= 1$
4 **for** $m = 2$ **to** n
5   **if** $s[m] \geq f[k]$
6         $A = A \cup \{a_m\}$
7         $k = m$
8 **return** A

Fractional Knapsack Problem - Pseudocode

```
Algorithm: Greedy-Fractional-Knapsack (w[1..n], p[1..n], W)
for i = 1 to n
   do x[i] = 0
weight = 0
for i = 1 to n
   if weight + w[i] ≤ W then
      x[i] = 1
      weight = weight + w[i]
   else
      x[i] = (W - weight) / w[i]
      weight = W
      break
return x
```

Huffman Coding Algorithm

Algorithm 7.6 HUFFMAN

Input: A set $C = \{c_1, c_2, \ldots, c_n\}$ of n characters and their frequencies $\{f(c_1), f(c_2), \ldots, f(c_n)\}$.

Output: A Huffman tree (V, T) for C.

Insert all characters into a min-heap H according to their frequencies.
$V \leftarrow C; \quad T = \{\}$
for $j \leftarrow 1$ **to** $n - 1$
$c \leftarrow$ DELETEMIN(H)
$c' \leftarrow$ DELETEMIN(H)
$f(v) \leftarrow f(c) + f(c')$ $\quad\{v$ is a new node$\}$
INSERT(H, v)
$V = V \cup \{v\}$ $\quad\{$Add v to $V\}$
$T = T \cup \{(v, c), (v, c')\}$ $\quad\{$Make c and c' children of v in $T\}$
end while

Dijkstra's Algorithm-Pseudocode

Algorithm 7.1 DIJKSTRA

Input: A weighted directed graph $G = (V, E)$, where $V = \{1, 2, \ldots, n\}$.

Output: The distance from vertex 1 to every other vertex in G.

$X = \{1\}; \quad Y \leftarrow V - \{1\}; \quad \lambda[1] \leftarrow 0$
for $y \leftarrow 2$ **to** n
if y is adjacent to 1 **then** $\lambda[y] \leftarrow length[1, y]$
else $\lambda[y] \leftarrow \infty$
end if
end for
for $j \leftarrow 1$ **to** $n - 1$
Let $y \in Y$ be such that $\lambda[y]$ is minimum
$X \leftarrow X \cup \{y\}$ $\quad\{$add vertex y to $X\}$
$Y \leftarrow Y - \{y\}$ $\quad\{$delete vertex y from $Y\}$
for each edge (y, w)
if $w \in Y$ **and** $\lambda[y] + length[y, w] < \lambda[w]$ **then**
$\lambda[w] \leftarrow \lambda[y] + length[y, w]$
end for
end for

Kruskal's Algorithm – Pseudocode

Algorithm 7.3 KRUSKAL

Input: A weighted connected undirected graph $G = (V, E)$ with n vertices.

Output: The set of edges T of a minimum cost spanning tree for G.

```
Sort the edges in E by nondecreasing weight.
for each vertex v ∈ V
    MAKESET({v})
end for
T = {}
while |T| < n − 1
    Let (x, y) be the next edge in E.
    if FIND(x) ≠ FIND(y) then
        Add (x, y) to T
        UNION(x, y)
    end if
end while
```