Python 3 Cheatsheet: Dynamic & Greedy

Dynamic Programming

Coin Change (Unbounded Knapsack)

```
def coinChange(coins, amount):
    dp = [float('inf')] * (amount + 1)
    dp[0] = 0
    for c in coins:
        for a in range(c, amount + 1):
            dp[a] = min(dp[a], dp[a - c] + 1)
    return dp[amount] if dp[amount] != float('inf') else -1
```

Longest Common Subsequence (2D DP)

```
def lcs(text1, text2):
    dp = [[0] * (len(text2) + 1) for _ in range(len(text1) + 1)]
    for i in range(len(text1)):
        for j in range(len(text2)):
            if text1[i] == text2[j]:
                dp[i+1][j+1] = 1 + dp[i][j]
            else:
                dp[i+1][j+1] = max(dp[i][j+1], dp[i+1][j])
    return dp[-1][-1]
```

Stock Profit (State Machine)

```
# T0: No stock, T1: Have stock
t0, t1 = 0, float('-inf')
for p in prices:
    t0 = max(t0, t1 + p) # Sell
    t1 = max(t1, t0 - p) # Buy
return t0
```

Greedy

Activity Selection / Merge Intervals Sort by end time or start time.

```
# Merge Intervals
intervals.sort()
merged = [intervals[0]]
for start, end in intervals[1:]:
```

```
        lastEnd = merged[-1][1]
        if start <= lastEnd:
            merged[-1][1] = max(lastEnd, end)
        else:
            merged.append([start, end])
```

Kadane's Algorithm

Max Subarray Sum.

```
def maxSubArray(nums):
    curSum = maxSum = nums[0]
    for n in nums[1:]:
        curSum = max(n, curSum + n)
        maxSum = max(maxSum, curSum)
    return maxSum
```