Homework Assignment $\lfloor \pi \rfloor$

Santa Claus
123456789

Smarch 32, 2017

Problem 1

```
procedure MaxSubarray(A)
    n = length(A)
    lb = ub = 1
    max = A[1]

    for i=1 to n do
        candidateMax = 0

        for j=i to n do
            candidateMax = candidateMax + A[j]

            if candidateMax > max then
                max = candidateMax
                lb = i
                ub = j
            end if
        end for
    end for

    return (lb,ub,max)
end procedure
```

Problem 2

When all elements of the array A are negative, the algorithm still works. It will return $(i, i, A[i])$, where A[i] is the greatest element or least negative element.

Problem 3

```
 1: procedure STRASSEN(A,B)
 2:     n = A.rows
 3:     C = new Matrix(n,n)
 4:
 5:     if n==1 then
 6:         C[1,1] = A[1,1] · B[1,1]
 7:     else
 8:         A11 = A[1...n/2, 1...n/2]                ▷ Upper-Left Quarter of Matrix A
 9:         A12 = A[n/2 + 1...n, 1...n/2]            ▷ Upper-Right Quarter of Matrix A
10:         A21 = A[1...n/2, n/2 + 1...n]            ▷ Bottom-Left Quarter of Matrix A
11:         A22 = A[n/2 + 1...n, n/2 + 1...n]        ▷ Bottom-Right Quarter of Matrix A
12:
13:         ...                  ▷ Repeat lines 8-11 for Matrix B instead of Matrix A
14:
15:         S1 = B12 - B22
16:         S2 = A11 + A12
17:         S3 = A21 + A22
18:         S4 = B21 - B11
19:         S5 = A11 + A22
20:         S6 = B11 + B22
21:         S7 = A12 - A22
22:         S8 = B21 + B22
23:         S9 = A11 - A21
24:         S10 = B11 + B12
25:
26:         P1 = STRASSEN(A11,S1)
27:         P2 = STRASSEN(S2,B22)
28:         P3 = STRASSEN(S3,B11)
29:         P4 = STRASSEN(A22,S4)
30:         P5 = STRASSEN(S5,S6)
31:         P6 = STRASSEN(S7,S8)
32:         P7 = STRASSEN(S9,S10)
33:
34:         C11 = P5 + P4 - P2 + P6
35:         C12 = P1 + P2
36:         C21 = P3 + P4
37:         C22 = P5 + P1 - P3 - P7
38:
39:         C[1...n/2, 1...n/2] = C11                ▷ Upper-Left Quarter of Matrix C
40:         C[n/2 + 1...n, 1...n/2] = C12            ▷ Upper-Right Quarter of Matrix C
41:         C[1...n/2, n/2 + 1...n] = C21            ▷ Bottom-Left Quarter of Matrix C
42:         C[n/2 + 1...n, n/2 + 1...n] = C22        ▷ Bottom-Right Quarter of Matrix C
43:     end if
44:
45:     return C
46: end procedure
```

Problem 4

$$
\begin{aligned}
C &= AB \\
&= \begin{bmatrix} 1 & 3 \\ 7 & 5 \end{bmatrix} \begin{bmatrix} 6 & 8 \\ 4 & 2 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
S1 &= B12 - B22 = (8) - (2) = 6 \\
S2 &= A11 + A12 = (1) + (3) = 4 \\
S3 &= A21 + A22 = (7) + (5) = 12 \\
S4 &= B21 - B11 = (4) - (6) = -2 \\
S5 &= A11 + A22 = (1) + (5) = 6 \\
S6 &= B11 + B22 = (6) + (2) = 8 \\
S7 &= A12 - A22 = (3) - (5) = -2 \\
S8 &= B21 + B22 = (4) + (2) = 6 \\
S9 &= A11 - A21 = (1) - (7) = -6 \\
S10 &= B11 + B12 = (6) + (8) = 14
\end{aligned}
$$

$$
\begin{aligned}
P1 &= Strassen(A11, S1) = 1 \cdot 6 = 6 \\
P2 &= Strassen(S2, B22) = 4 \cdot 2 = 8 \\
P3 &= Strassen(S3, B11) = 12 \cdot 6 = 72 \\
P4 &= Strassen(A22, S4) = 5 \cdot -2 = -10 \\
P5 &= Strassen(S5, S6) = 6 \cdot 8 = 48 \\
P6 &= Strassen(S7, S8) = -2 \cdot 6 = -12 \\
P7 &= Strassen(S9, S10) = -6 \cdot 14 = -84
\end{aligned}
$$

$$
\begin{aligned}
C11 &= P5 + P4 - P2 + P6 \\
&= (48) + (-10) - (8) + (-12) \\
&= 18 \\
C12 &= P1 + P2 \\
&= (6) + (8) \\
&= 14 \\
C21 &= P3 + P4 \\
&= (72) + (-10) \\
&= 62 \\
C22 &= P5 + P1 - P3 - P7 \\
&= (48) + (6) - (72) - (-84) \\
&= 66
\end{aligned}
$$

$$
C = \begin{bmatrix} 18 & 14 \\ 62 & 66 \end{bmatrix}
$$

Problem 5

Problem 6

Schedule the jobs such that the jobs with the smallest run time will occur first. A given job will have its completion time be the sum of running times for jobs before and its own job.

$$\text{Completion Time}(j_i) = \sum_{k=1}^{i} t_k$$

In other words, subsequent jobs will always have an addition to its runtime. However, we can minimize the runtime by always adding the smallest values.

Suppose job running times are sorted such that:

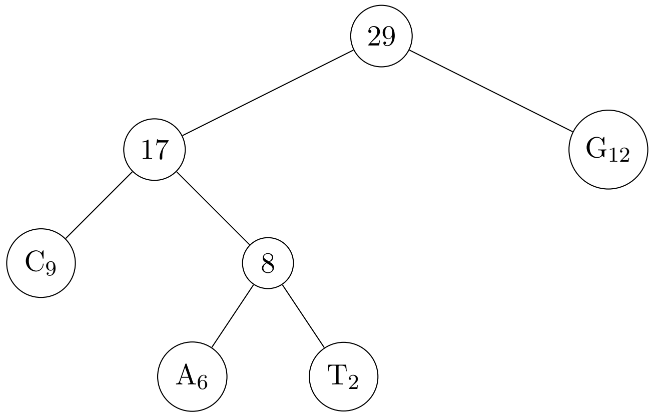
$$s_0 \leq s_1 \leq ... \leq s_n$$

$$\begin{aligned}
\text{Average Run Time} &= \frac{1}{n}\left[(s_1) + (s_1 + s_2) + (s_1 + s_2 + s_3)...(s_1 + s_2 + ... + s_n)\right] \\
&= \frac{1}{n}\left[(n)s_1 + (n-1)s_2 + (n-2)s_3 + ... + s_n\right] \\
&= \frac{1}{n}\sum_{i=1}^{n}(n+1-i)s_i \\
&= \frac{1}{n}\sum_{i=1}^{n}(n+1)s_i - \frac{1}{n}\sum_{i=1}^{n} i \cdot s_i \\
&= (1+\frac{1}{n})\sum_{i=1}^{n} s_i - \frac{1}{n}\sum_{i=1}^{n} i \cdot s_i
\end{aligned}$$

$(1+\frac{1}{n})\sum_{i=1}^{n} s_i$ is an expression that is independent of ordering. It can be seen as the base cost of running all the jobs. The second term, however, is not independent of ordering. $\frac{1}{n}\sum_{i=1}^{n} i \cdot s_i$ shows that an earlier job has a smaller constant while subsequent jobs will have a larger constant.

Example: The second summation contains $1 \cdot s_1$ and $2 \cdot s_2$. Suppose $s_1 > s_2$, then the overall summation will be less than when $s_1 < s_2$. If the overall summation is smaller, then the average running time has now increased.

Problem 7



letter	frequency	codeword	bits needed
A	6	010	3
C	9	00	2
G	12	1	1
T	2	011	3

Original Text = ACCGGTCGAGTGCGCGGAAGCCGGCCGAA
Encoded String = 010000011011001010101110010011010010100001100001010010
Bits Needed = 54

Problem 8