Midterm

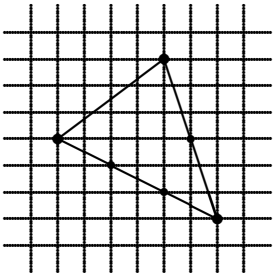
Note

This is a closed-book exam. Each problem accounts for 10 points, unless otherwise marked.

Problems

1. Construct a Gray code of length $\lceil \log_2 14 \rceil$ $(= 4)$ for 14 objects. Please explain how the Gray code is constructed *systematically* from Gray codes of smaller lengths.

2. The *lattice* points in the plane are the points with integer coordinates. Let T be a triangle such that all of its three vertices are lattice points; see the figure below. Let p be the number of lattice points that are on the boundary of T (including its vertices), and let q be the number of lattice points that are inside T. Prove that the area of T is $\frac{p}{2} + q - 1$.



3. Consider labeling the nodes of a full binary tree level by level, from top to bottom and left to right, with the numbers 1 through n, where n is the number of nodes in the tree. Prove *by induction* that, for an internal node labeled i, its left and right children are labeled $2i$ and $2i + 1$ respectively.

4. Consider the problem of merging two skylines, which is a useful building block for computing the skyline of a number of buildings. A skyline is an alternating sequence of x coordinates and y coordinates (heights), ending with an x coordinate (as discussed in class). The sequence of coordinates may be coveniently stored in an array, say A, with $A[0]$ storing the first x coordinate, $A[1]$ the first y coordinate, $A[2]$ the second x coordinate, etc.

   Design a linear-time procedure that prints out the resulting skyline from merging two given skylines. Please present the procedure in suitable pseudocode. The procedure should be named `merge_skylines` and invoked by `merge_skylines(A,m,B,n)`, where `A` and `B` are the two input skylines and `A[m]` and `B[n]` store the final x coordinate of skyline `A` and that of skyline `B` respectively. Does your procedure really run in $O(m + n)$ time? Please explain.

5. Below is the pseudocode of the binary search algorithm we discussed in class. Would the code still be correct if we change the assignment "$Middle := \lceil \frac{Left+Right}{2} \rceil$" to "$Middle := \lfloor \frac{Left+Right}{2} \rfloor$" for $Middle$ to take instead the largest integer less than or equal to $\frac{Left+Right}{2}$? Please justify your answer.

```
function Find (z, Left, Right) : integer;
begin
    if Left = Right then
       if X[Left] = z then Find := Left
       else Find := 0
    else
       Middle := ⌈(Left+Right)/2⌉;
       if z < X[Middle] then
          Find := Find(z, Left, Middle − 1)
       else
          Find := Find(z, Middle, Right)
end

Algorithm Binary_Search (X, n, z);
begin
    Position := Find(z, 1, n);
end
```

6. Given the array below as input, what are the contents of array *TEMP* after the merge part is executed for the first time and what are the contents of *TEMP* when the algorithm terminates? Assume that each entry of *TEMP* has been initialized to 0 when the algorithm starts.

1	2	3	4	5	6	7	8	9	10	11	12
7	9	2	6	5	10	8	3	1	12	4	11

7. Consider rearranging the following array into a max heap using the *bottom-up* approach.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
4	2	8	5	1	14	7	6	3	11	10	13	15	12	9

Please show the result (i.e., the contents of the array) after a new element is added to the current collection of heaps (at the bottom) until the entire array has become a heap.

8. Draw a decision tree of the Heapsort algorithm (in increasing order) for the case of $A[1..3]$, i.e., $n = 3$. In the decision tree, you must indicate (1) which two elements of the original input array are compared in each internal node and (2) the sorting result in each leaf. Please use X_1, X_2, X_3 (not $A[1]$, $A[2]$, $A[3]$) to refer to the elements (in this order) of the original input array A.

9. The *next* table is a precomputed table (for $B = b_1 b_2 \cdots b_m$) that plays a critical role in the KMP algorithm. Under what condition regarding $b_1 b_2 \cdots b_i$, $2 \leq i \leq m$, will $next[i]$ get a 0 in the preprocessing? And under what condition can it be safely set to -1 (without missing a potential match when searching for B in another input string)?

10. Given two strings $A = bbaaba$ and $B = ababa$, what is the result of the minimal cost matrix $C[0..6, 0..5]$, according to the algorithm discussed in class for changing A character by character into B? Aside from giving the cost matrix, please show the details of how the entry $C[4, 3]$ is computed from the values of $C[3, 2]$, $C[3, 3]$, and $C[4, 2]$.

Appendix

- The Mergesort algorithm:

```
Algorithm Mergesort (X, n);
begin  M_Sort(1, n)  end

procedure M_Sort (Left, Right);
begin
    if Right − Left = 1 then
       if X[Left] > X[Right] then swap(X[Left], X[Right])
    else if Left ≠ Right then
            Middle := ⌈½(Left + Right)⌉;
            M_Sort(Left, Middle − 1);
            M_Sort(Middle, Right);
            // the merge part
            i := Left;  j := Middle;  k := 0;
            while (i ≤ Middle − 1) and (j ≤ Right) do
                  k := k + 1;
                  if X[i] ≤ X[j] then
                     TEMP[k] := X[i];  i := i + 1
                  else TEMP[k] := X[j];  j := j + 1;
            if j > Right then
               for t := 0 to Middle − 1 − i do
                   X[Right − t] := X[Middle − 1 − t]
            for t := 0 to k − 1 do
                X[Left + t] := TEMP[1 + t]
end
```

- The KMP algorithm (assuming *next*):

```
Algorithm String_Match (A, n, B, m);
begin
    j := 1;  i := 1;
    Start := 0;
    while Start = 0 and i ≤ n do
        if B[j] = A[i] then
            j := j + 1;  i := i + 1
        else
            j := next[j] + 1;
            if j = 0 then
```

```
                j := 1;  i := i + 1;
        if j = m + 1 then Start := i − m
end
```

- The algorithm for computing the *next* table in the KMP algorithm:

```
Algorithm Compute_Next (B, m);
begin
    next[1] := −1;  next[2] := 0;
    for i := 3 to m do
        j := next[i − 1] + 1;
        while B[i − 1] ≠ B[j] and j > 0 do
            j := next[j] + 1;
        next[i] := j
end
```