1. Fill in the blanks.

(i) [10 points] The following is pseudo-code for what algorithm?

```
int x[n];
obtain values of x;
for(int i = n-1; i > 0; i++)
  for(int j = 0; j < i; j++)
    if(x[i] < x[j]) swap(x[i],x[j]);
```

(ii) [10 points] ________________ algorithm does not allow the weight of any arc to be negative.

(iii) [10 points] The time complexity of every comparison-based sorting algorithm is ________________.
    (Your answer should use \( \Omega \) notation.)

(iv) [10 points] The prefix expression \(*a + b \sim c d \sim e\) is equivalent to the infix expression
    ________________ and the postfix expression ________________.

(v) [10 points] The items stored in a priority queue represent ________________ ________________.

(vi) [10 points] The asymptotic complexity of Dijkstra’s algorithm algorithm is ________________.

(vii) [10 points] A ________________ hash function fills the hash table exactly with no collisions.

(viii) [10 points] In ________________ ________________ there can be any number of items at a given index of the hash table.

(ix) [10 points] If the position at \( h(x) \) is already occupied for some data item \( x \), a ________________ ________________ is used to find an unoccupied position in the hash table.

(x) [10 points] In ________________ hashing, each item has more than one hash value, but only uses one of them.

(xi) [10 points] If \( G \) is a weighted directed graph, then it is impossible to solve the all pairs shortest path problem for \( G \) if \( G \) has a ________________ ________________.
(xii) [10 points] A planar graph with $n$ vertices can have no more than ________ edges. (Exact formula, please.)

2. Give the asymptotic complexity, in terms of $n$, of each of the following code fragments. [10 points each]
   (a) 
   ```cpp
   for(int i = 2; i < n; i = i*i)
       cout << "Hello world" << endl;
   ```
   
   (b) 
   ```cpp
   for(int i = 1; i < n; i++)
       for(int j = 1; j < i; j = 2*j)
           cout << "Hello world" << endl;
   ```
   
   (c) 
   ```cpp
   for(int i = 1; i*i < n; i++)
       cout << "hello world" << endl;
   ```
   
   (d) 
   ```cpp
   for(int i = 0; i < n; i++)
       for(int j = n; j > i; j = j/2)
           cout << "Hello world!" << endl;
   ```
   
   (e) 
   ```cpp
   for(int i = 2; i < n; i=i*i)
       cout << "Hello world!" << endl;
   ```
   
   (f) 
   ```cpp
   for(int i = 1; i*i < n; i++)
       cout << "Hello world!" << endl;
   ```
   
   (g) 
   ```cpp
   for(int i = 1; i < n; i++)
       for(int j = 2; j < i; j=j*j)
           cout << "Hello world" << endl;
   ```

3. Solve the recurrences. Give the asymptotic value of $F(n)$ in terms of $n$, using $\Theta$ notation. [10 points each]
   (a) $F(n) = 4F\left(\frac{n}{2}\right) + n$

   (b) $F(n) \geq F\left(\frac{n}{2}\right) + 2F\left(\frac{n}{4}\right) + n$

   (c) $F(n) \leq 2F\left(\frac{n}{2}\right) + n^2$

   (d) $F(n) \geq 3F\left(\frac{n}{9}\right) + 1$

   (e) $F(n) = F\left(\frac{3n}{5}\right) + 4F\left(\frac{2n}{5}\right) + n^2$

   (f) $F(n) \leq 4F\left(\frac{n}{2}\right) + n^2$
(g) $F(n) \leq F(\sqrt{n}) + 1$

4. [20 points] List properties of a good hash function used for a search structure.

5. [20 points] Find an optimal prefix code for the alphabet \{A, B, C, D, E, F\} where the frequencies are given in the following array.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>4</td>
</tr>
<tr>
<td>B</td>
<td>5</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>6</td>
</tr>
<tr>
<td>E</td>
<td>8</td>
</tr>
<tr>
<td>F</td>
<td>1</td>
</tr>
</tbody>
</table>

7. [20 points] Use the DFS method to find the strong components of the digraph in the first figure below. Show steps, using the second figure as needed.

![Graph Diagram](image)

8. [20 points] What is the loop invariant of the loop in the following function?

```c
float product(float x, int n)
{
    float z = 0.0;
    float y = x;
    int m = n;
    while(m > 0)
    {
        if(m%2) z = z+y;
        m = m/2;
        y = y+y;
    }
    return z;
}
```

9. [20 points] Write pseudocode for the Bellman-Ford algorithm. Be sure to include the shortcut that ends the program when the final values have been found.
10. Consider the following C++ code.

```cpp
int george(int n)
{
    if(n == 0) return 1;
    else return george(n/2)+george(n/2-1)+n*n;
}
```

(a) [10 points] What is the asymptotic complexity of \texttt{george(n)}?

(b) [10 points] What is the time complexity of the recursive code given above?

(c) [10 points] What is the time complexity of a dynamic programming algorithm to compute \texttt{george(n)}?

(d) [10 points] What is the space complexity of a computation of \texttt{george(n)} using memoization?

11. [20 points] Walk through mergesort with the array given below.

\texttt{VJATNLDQMEFSPWGL}
12. [20 points] Write pseudocode for the simple coin-row problem we discussed in class. You are given a row of $n$ coins of various values. The problem is to select a set of coins of maximum total value, subject to the condition that no two adjacent coins are selected. Your code should identify the coins which are selected.

13. [20 points] Write pseudocode for a function $\text{float power(float } x, \text{ int } n)$ that returns $x^n$. You may assume that $x \neq 0$, but your code must work for any integer $n$. It is not necessary to use the algorithm given in class; use any algorithm that works.
14. [20 points] The following code is used as a subroutine for both quicksort and select. Assume A[n] is an array of integers. For simplicity, we assume that no two entries of A are equal. Write a loop invariant for the while loop.

```c
int pivot = A[0];
int lo = 0;
int hi = n-1;
while(lo < hi)
{
    if(A[lo+1] < pivot) lo++;
    else if(A[hi] > pivot) hi--;
    else swap(A[lo+1],A[hi]);
}
```