**University of Nevada, Las Vegas Computer Science 477/677 Spring 2025**

**Answers to Assignment 3: Due Saturday February 8, 2025**

1. Give the names of three kinds of priority queue.

   **stack**, **queue**, **heap**

2. In any programming application, The items of a priority queue represent **unfulfilled obligations**.

3. *Pop* and *push* are operators of the ADT **stack**.

4. *Fetch* and *store* are operators of the ADT **array**.

5. The answer to each of these questions is either *bubbleup* or *bubbledown*.

   If a heap is implemented a binary tree:

   (a) Insertion into the heap requires the use of **bubbleup**.

   (b) Deletmax (or deletemin, as the case may be) reuquires use of **bubbledown**.

6. The following is C++ code for which quadratic time sorting algorithm?

   **selection sort**

```
int x[N];

void swap(int x, int y)
 {
  int temp = x;
  x = y;
  y = temp;
 }

void sort()
 {
  for(int i = 0; i < N; i++)
   for(int j = i+1; j < N; j++)
    if(x[j] < x[i]) swap(x[i],x[j]);
 }
```

7. The following is C++ code for a function that computes a floating point number to the power of a positive integer. Find a loop invariant which can be used to prove correctness of the function.

```
float power(float x, int n) // input condition: n > 0
 {
  assert(n > 0);
  float z = 1.0;
  float y = x;
  int m = n;
  while(m > 0)
   {
    if(m%2) z = y*z;
    y = y*y;
    m = m/2;
   }
  return z;
 }
```

The function computes $x^n$. The loop invariant is:

**I will post the answer later.**

8. We can experimentally analyze the time complexity of code by using a counter. In the following example, we count the number of iterations of the loop.

```
cout << "Enter a positive integer" << endl;
int n;
cin >> n;
int kount = 0;
for(int i = 0; i < n; i++)
 kount++;
cout << "kount = " << kount << endl;
```

Of course, the output will be n. We can say that the time complexity of the code is $\Theta(n)$.

For each of the code fragments below, find the asymptotic time complexity in terms of n. Assume that the value of n is given.

(a)
```
for(int i = 1; i < n; i=2*i)
```
$\Theta(\log n)$

(b)
```
for(int i = n; i > 1; i=i/2)
```
$\Theta(\log n)$

(c)
```
for(int i = 1; i < n; i++)
  for(int j = 1; j < i; j = j*2)
```
$\Theta(n \log n)$

(d)
```
for(int i = 1; i < n; i++)
  for(int j = i; j < n; j = j*2)
```
$\Theta(n)$

(e)
```
for(int i = 1; i*i < n; i++)
```
$\Theta(\sqrt{n})$

9. Create a treap with the following itmes, each with the priority shown in the table. Use min-heap order.

| item | priority |
|---|---|
| V | 3 |
| A | 5 |
| S | 6 |
| Q | 4 |
| N | 7 |
| K | 2 |