University of Nevada, Las Vegas Computer Science 477/677 Fall 2022

Assignment 2: Due Saturday February 4 2023 11:59 PM

Name:_____

You are permitted to work in groups, get help from others, read books, and use the internet. Your answers must be written in a pdf file and uploaded to canvas, by midnight September 9th. Your file must not be unnecessarily long. If you have any questions, or you are having trouble uploading the assignment you may email the grader, Sepideh Farivar, at `farivar@unlv.nevada.edu`. You may also send me email to ask questions.

1. Fill in the blanks. The answer to each question is either "insertion sort," "selection sort," or "bubblesort."

   (a) Treesort is a form of _____.

   (b) Heapsort is a form of _____.

   (c) The following C++ code implements _____.

```
int x[N];

void swap(int&a,int&b)
 {
  int temp = a;
  a = b;
  b = 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]);
 }
```

2. Use a combination of the methods shown in class to find the asymptotic complexity of each of these code fragments, in terms of n. Express the answers using Θ notation.

   (a)
```
for(int i = 2; i < n; i=i*i)
 for(int j = i; j < n; j++)
  cout << i << " " << j << endl;
```

   (b)
```
for(int i = 1; i*i < n; i=2*i)
 cout << i << endl;
```

3. Below is an array implementation of a stack type in C++. Fill in the missing parts of the functions `empty` and `pop`. Assume that N is large enough to prevent overflow.

```
struct stackoffloat
 {
  float item[N];
  int size;
 };

void klear(stackoffloat&s)
 // make s empty
 {
  s.size = 0;
 }

void push(float x, stackoffloat&s)
 // insert a new item onto the top of s
 {
  s.item[s.size] = x;
  s.size++;
 }

bool empty(stackoffloat&s)
 // true if s is empty, false otherwise
 {




 }

float pop(stackoffloat&s)
 // return and delete the top item of s
 {




 }
```

4. Below is an array implementation of a queue type in C++. Fill in the missing parts of the functions `empty` and `dequeue`. Assume that N is large enough to prevent overflow.

```
struct queueoffloat
 {
  float item[N];
  int front;
  int rear;
 };

void klear(queueoffloat&q)
 // make q empty
 {
  q.front = 0;
  q.rear = -1;
 }

void enqueue(float x, queueoffloat&q)
 // insert a new item at the rear of q
 {
  q.rear++;
  q.item[q.rear] = x;
 }

bool empty(queueoffloat&q)
 // true if q is empty, false otherwise
 {









 }

float dequeue(queueoffloat&q)
 // return and delete the front item of q
 {









 }
```

5. Write the loop invariant for the loop in the C++ code below.

```
bool isprime(int n)
 // input condition: n > 1
 {
  bool prime = true;
  int d = 1;
  while(prime and d < n-1)
   {
    d++;
    prime = n%d;
   }
  return prime;
 }
```

6. Walk through polyphase mergesort for the following list.

   QYJERGBFPWDAUVCXH