Homework # 1

CS4040/5040 (100/110 pts)

Due Friday, September 10, 2021 by midnight.

1. (60 pts.) Analyzing Code. For each of the following, answer parts a-e.

   a.) (5pts.) Is the code correct? If not, provide a counterexample.

   b.) (3pts.) What is the asymptotic worst-case time complexity of the code.

   c.) (2pts.) What is the asymptotic worst-case space complexity of the code.

   d.) (3pts.) What is the asymptotic best-case time complexity of the code.

   e.) (2pts.) What is the asymptotic best-case space complexity of the code.

   i.) (15pts.) Maximum element (note this code is not the same as in the example in `Notes3`)

```
double Max(double a[], int b, int e)
{
  if (b>=e) {
    return a[b];
  } else {
    int m=(b+e)/2;
    return(fmax(Max(a, b, m), Max(a, m+1, e)));
  }
}

// The following is the top level call to Max.
// Returns the largest element in a. There are n elements in a[]

double Max(double a[], int n)
{
  return(Max(a, 0, n-1));
}
```

   ii.) (15pts.) Sequential Search version 1

```
%% Determines if element x is in array a, by looking at the
%% first 1/3, then the middle 1/3, finally at the last 1/3.
%%
int Find(double x, double a[], int b, int e) {
int n=e-b+1;
switch (n) {
case 0:
case 1:
   return((a[e]==x)?e+1:0);
case 2:
   return((a[b]==x)?(b+1):((a[e]==x)?(e+1):0));
}
int m1=b+n/3;
int m2=b+(2*n)/3;
int res1=Find(x, a, b, m1);
if (res1==0) {
    int res2=Find(x, a, m1+1, m2);
    if (res2==0) {
      int res3=Find(x, a, m2+1, e);
      if (res3==0) {
         return(0);
      } else return(res3);
    } else return(res2);
  } else return(res1);
}
// The following is the top level call to Find.
```

```
// Returns 0 if x is not in a, or index between 1 and n if it is.
// There are n elements in a.
int Find(double x, double a[], int n) {
  return(Find(x, a, 0, n-1));
}
```

iii.) (15pts.) Sequential Search version 2

```
int Find2(double x, double a[], int b, int e)
{
  if (b>=e){
    return(0);
  }
  if (a[b]==x) {
    return(b+1);
  } else {
    return(Find2(x,a,b+1,e));
  }
}

// The following is the top level call to Find2.
// Returns 0 if x is not in a, or index between 1 and n if it is.
// There are n elements in a.

int Find2(double x, double a[], int n)
{
  return(Find2(x, a, 0, n-1));
}
```

iv.) (15pts.) Do two arrays hold the same elements (possibly in a different order.)

```
// Returns true if the two arrays hold the exact same
// elements (not necessarily in the same order).
//
// We may assume that the two arrays are of the same size,
// otherwise they cannot possibly hold the same elements.

bool EqualSets(int a[], int b[], int n)
{
  for (int i=0;i<n;++i){
    bool found=false;
    for (int j=0;j<n;++j) {
      if (a[i]==b[j]) {
        found = true;
        break;
      }
    }
    if (!found) {
      return(false);
    }
  }
  return(true);
}
```

2. (30 pts.) Prove or disprove by using the definition that:

   (a) (10 pts.) $2n \in O(6n^2 + n + 42)$

   (b) (10 pts.) $2n^2 \in o(3n^2 + 12n)$

   (c) (10 pts.) $3n^2 + 42 \in \Omega(4n + 1)$

3. (10 pts.) Answer true or false to each of the following. You do not have to provide justification.

   (a) $10 + n + n^2 \in O(n^2)$

(b) $10 + n + n^2 \in \Theta(n^2)$

(c) $10 + n + n^2 \in \Omega(n)$

(d) $10 + n + n^2 \in \Theta(n)$

(e) $10 + n + \log n \in \Omega(n)$

(f) $10 + n + \log n \in O(n^2)$

(g) $7 \log^2 n + 2n \log n \in O(n)$

(h) $7 \log^2 n + 2n \log n \in \Omega(\log n)$

(i) $\left(\frac{1}{3}\right)^n + 100 \in O(1)$

(j) $3^n + 100 \in O(1)$

CS5040 students must also do the following problem

4. (10 pts.) Exercise 2-4, page 41