Tölvunarfræði 2
Miðannarpróf
Lausn

Allar spurningar gilda 12.5%. Bestu 8 svör gilda til einkunnar. Engin hjálpargögn eru leyfileg. Merkið svarblöð með **nafni** og **númeri dæmahóps** (D1, D2, D3, D4 eða D5).

All questions count 12.5% The best 8 answers count toward the grade. No help materials are allowed. Mark all answer sheet with your **name** and the number of your **exercise group** (D1, D2, D3, D4 or D5).

Nafn/Name:_____

Dæmahópur/Exercise group:_____

1. Fyllið inn þar sem spurningarmerkin eru, þ.e. skrifið hvað á að koma í stað `?1?`, o.s.frv. Þetta eru samtals átta svör.

   Fill in where the question marks are, i.e. write what should replace `?1?`, etc. This is a total of eight answers.

```
// Notkun: k = leita(f,i,j,x);
// Fyrir:  f[i..j-1] er í minnkandi röð
// Eftir:  f[i..j-1] er óbreytt, i-1 <= k <= j-1, og
//         f[i..k] >= x > f[k+1..j-1]

// Usage:  k = leita(f,i,j,x);
// Pre:    f[i..j-1] is in descending order
// Post:   f[i..j-1] is unchanged, i-1 <= k <= j-1, and
//         f[i..k] >= x > f[k+1..j-1]

int leita( double[] f, int i, int j, double x ) {
  if( i==j ) return i-1;
  int m = (i+j)/2;
  if( f[m] ?1? x )
    return leita(f,i,?2?,x);
  else
    return leita(f,?3?,j,x);
}
```

```
int leita( double[] f, int i, int j, double x ) {
  int p=i, q=j;
  while( ?4? ) {
    // | >=x | óþekkt | <x |
    //   i      p         q     j
    //
    // | >=x | unknown | <x |
    //   i      p          q     j
    int m = (?5?)/2;
    if( f[m] ?6? x )
      p = ?7?;
    else
      q = ?8?;
  }
  return p-1;
}
```

Svar/Answer:

?1?: <

?2?: m

?3?: m+1

?4?: p!=q

?5?: p+q

?6?: >=

?7?: m+1

?8?: m

2. Gerið ráð fyrir að einhver annar hafi skrifað fall `leita`, sem hefur lýsingu eins og að ofan. Skrifið fall, ásamt lýsingu, sem notar fallið `leita` til að skila boolean gildi (true eða false), sem segir til um hvort tiltekið fylki double talna, sem raðað er fyrirfram í minnkandi röð, inniheldur tiltekna double tölu. Fylkið og talan sem leitað er að skulu vera viðföng í fallið ykkar. Vísbending: Lengd fylkis `f` er `f.length`.

   Assume someone else has written a function `leita`, which has a description as above. Write a function, with a description, that uses the function `leita` to return a boolean value (true or false), that indicates whether a given array of doubles, that is in descending order, contains a given double number. The array and the number being searched for should be arguments to your funciton. Hint: the length of an array `f` is `f.length`.

   Svar/Answer:

```
// Notkun: b = has(f,x);
// Fyrir:  f er í minnkandi röð.
// Eftir:  b er satt þþaa x sé í f.
```

```
static boolean has( double[] f, double x )
{
  int k = leita(f,0,f.length,x);
  if( k < 0 ) return false;
  return f[k]==x;
}
```

3. Fyllið inn þar sem spurningarmerkin eru. Þetta er insertion sort.

   Fill in where the question marks are. This is insertion sort.

```
// Notkun: swap(f,i,j);
// Fyrir:  f[i] og f[j] eru sæti í f
// Eftir:  Búið er að víxla gildunum í f[i] og
//         f[j]

// Usage:  swap(f,i,j);
// Pre:    f[i] and f[j] are positions in f
// Post:   The values in f[i] and f[j] have
//         been swapped.

void swap( double[] f, int i, int j )
{
  double tmp = f[i];
  f[i] = f[j];
  f[j] = tmp;
}

// Notkun: sort(f,i,j);
// Fyrir:  f[i..j-1] er ekki-tómt svæði í f
// Eftir:  Búið er að raða svæðinu f[i..j-1]

// Usage:  sort(f,i,j);
// Pre:    f[i..j-1] is a non-empty area of f
// Post:   The area f[i..j-1] has been sorted

void sort( double[] f, int i, int j ) {
  int p = ?1?;
  while( ?2? ) {
    // | í vaxandi röð | óþekkt |
    //  i              p       j
    //
    // | in ascending order | unknown |
    //  i                    p         j
    //
    // i <= p <= j

    int k = ?3?;
```

```
      while( ?4? ) {
        // a) | næstum í vax. röð | óþekkt |
        //     ^         ^         ^          ^
        //     i         k         p          j
        //     svæðið sem er næstum í vaxandi röð
        //     er aðeins rangt að því leyti að f[k]
        //     er e.t.v. of aftarlega.
        //
        // a) | almost in asc. order | unknown |
        //     ^          ^          ^          ^
        //     i          k          p          j
        //     the area that is almost in ascending
        //     order is only wrong in that f[k] is
        //     perhaps too far to the right.
        // b) i <= k <= p < j
        swap(f,k,k-1);
        k--;
      }
      p++;
    }
  }
```

Svar/Answer:

?1?: i+1

?2?: p!=j

?3?: p

?4?: k!=i && f[k]<f[k-1]

4. Hver er fastayrðing lykkju í ytri lykkju selection sort?

   What is the loop invariant in the outer loop of selection sort?

   Svar/Answer:

```
| minnst í vaxandi röð | óþekkt |
i                      k        j
```

5. Hver tvö af eftirfarandi getur verið fastayrðing fyrri lykkju heapsort?

   Which two of the following can be the loop invariant of the first loop of heapsort?

   (a)
```
| minnst í vaxandi röð | hrúga (heap) |
0                      i              n
```

```
| smallest in ascending order | heap |
0                             i      n
```

(b)
```
| stærst í vaxandi röð | hrúga (heap) |
 0                     i              n

| biggest in ascending order | heap |
 0                           i      n
```

(c)
```
| óþekkt | uppfyllir hrúguskilyrði |
 0        i                        n

| unknown | fulfills heap condition |
 0         i                        n
```

(d)
```
| minnst í vaxandi röð | uppfyllir hrúguskilyrði |
 0                     i                          n

| smallest in ascending order | fulfills heap condition |
 0                             i                            n
```

(e)
```
| óþekkt | uppfyllir hrúguskilyrði |
 0        i                        n

| unknown | fulfills heap condition |
 0         i                        n
```

(f)
```
| hrúga (heap) | stærst í vaxandi röð |
 0              i                     n

| heap | biggest in ascending order |
 0      i                           n
```

(g)
```
| hrúga (heap) | minnst í vaxandi röð |
 0              i                     n

| heap | smallest in ascending order |
 0      i                            n
```

(h)
```
| hrúga (heap) | óþekkt |
 0              i        n

| heap | unknown |
 0      i         n
```

Svar/Answer:

(a) Nei

(b) Nei

(c) Já

(d) Nei

(e) Já

(f) Nei

(g) Nei

(h) Já

6. (a) Hver er fastayrðing seinni lykkju heapsort?
   What is the loop invariant of the second loop of heapsort?

   (b) Hver er tímaflækja heapsort?
   What is the time complexity of heapsort?

   Svar/Answer:

```
| hrúga | stærst í vaxandi röð |
 0       i                      n
```

   Tímaflækjan er $O(n \log n)$.

7. (a) Lýsið hugmyndinni í merge-sort.
   Describe the idea in merge-sort.

   (b) Hver er tímaflækja merge-sort?
   What is the time complexity of merge-sort?

   Svar/Answer:
   Ef runa inniheldur eitt eða færri gildi þarf ekkert að gera því runan er þá þegar í röð. Ef svo er ekki þá gerum við eftirfarandi.

   Skiptum fyrst gildunum í tvær jafnstórar runur (það má muna einum á fjölda gilda). Röðum síðan hverri runu fyrir sig og þegar því er lokið samröðum við röðuðu rununum í eina raðaða runu.

   Tímaflækjan er $O(n \log n)$.

8. Gefið er eftirfarandi stef.

   Given is the following function.

```
// Notkun:   k = skipta(f,i,j);
// Fyrir:    f[i..j-1] er ekki-tómt svæði í fylkinu f
// Eftir:    Búið er að víxla gildum í svæðinu og gefa
//           k gildi þ.a.
//            1) i <= k < j og
//            2) f[i..k-1] <= f[k] <= f[k+1..j-1]

// Usage:    k = skipta(f,i,j);
// Pre:      f[i..j-1] is a non-empty area of the array f
// Eftir:    Values in the area have been swapped and k
//           has received a value such that
//            1) i <= k < j and
//            2) f[i..k-1] <= f[k] <= f[k+1..j-1]

static int skipta( double f[], int i, int j ) {...}
```

Skrifið quicksort stef (með lýsingu - notkun, fyrir og eftir) með hjálp þessa stefs. Ekki þarf að forrita `skipta` stefið.

Write a quicksort function (with description - usage, pre and post) using this as a helper function. You do not need to program the `skipta` function.

Svar/Answer:

```
// Notkun: sort(f,i,j);
// Fyrir:  f[i..j-1] er svæði í f.
// Eftir:  Búið er að raða svæðinu í vaxandi röð.
static void sort( double[] f, int i, int j )
{
  if( j-i < 2 ) return;
  int k = skipta(f,i,j);
  sort(f,i,k);
  sort(f,k+1,j);
}
```

9. Hver er tímaflækja eftirfarandi röðunaraðferða? Hvers konar tímaflækju er um að ræða (versti tími eða meðaltími)? Munið að við höfum varla áhuga á besta tíma og við höfum áhuga á meðaltíma ef og aðeins ef hann er betri en versti tími.

   What is the time complexity of the following sorting algorithms? What kind of complexity is it (worst time, expected time, etc.)? Remember that we are hardly interested in the best time and we are only interested in the expected time if it is better than the worst time.

   (a) Insertion-sort
   (b) Selection-sort
   (c) Quicksort
   (d) Merge-sort
   (e) Radix-sort (gerið ráð fyrir þriggja stafa tölum – assume three digit numbers)
   (f) Heapsort

   Svar/Answer:

   Insertion-sort. $O(n^2)$ versti tími.

   Selection-sort. $O(n^2)$ versti tími.

   Quicksort. $O(n^2)$ versti tími, $O(n \log n)$ meðaltími.

   Merge-sort. $O(n \log n)$ versti tími.

   Radix-sort (gerið ráð fyrir þriggja stafa tölum – assume three digit numbers). $O(n)$ versti tími.

   Heapsort. $O(n \log n)$ versti tími.

10. Gefið er eftirfarandi fall.

    Given is the following function.

```
public static double pow( double x, int n )
{
  if( n==0 ) return 1.0;
  if( n==1 ) return x;
  if( n==2 ) return x*x;
  double a = pow(x,n/2);
  double b = pow(x,(n-1)/2);
  if( a!=b )
    return pow(x,n/2)*pow(x,n/2);
  else
    return x*pow(x,(n-1)/2)*pow(x,(n-1)/2);
}
```

    (a) Sýnið rakningarvensl fyrir $T(n)$, tímaflækju fallsins, sem fall af n, sem er seinna viðfang pow fallsins.

    Show a recurrence relation for $T(n)$, the time complexity of the function, as a function of n, which is the second argument to the pow function.

    (b) Hver er stærðargráða $T(n)$, þ.e. stærðargráða tímaflækju fallsins?

    What is the order of magnitude of $T(n)$, i.e. the order of the time complexity of the function?

    Svar/Answer:
    Rakningin er $T(n) = 4T(\frac{n}{2}) + O(1)$. Stærðargráðan er $T(n) = O(n^2)$.

11. Gerið ráð fyrir að skrifuð sé merge sort útfærsla sem raðar biðröðum og virkar þannig að óraðaðri biðröð er skipt í fimm jafnstórar biðraðir, þeim biðröðum er síðan raðað, og röðuðu biðröðunum er síðan samraðað í upphaflegu biðröðina.

    Skrifið rakningarvensl fyrir tímaflækju þessarar útfærslu og sýnið stærðargráðu tímaflækjunnar sem fall af n, þar sem n er fjöldi gilda í upphaflegu biðröðinni.

    Assume that someone programs a merge sort variant that sorts queues and works by splitting an unordered queue into five equally sized queues, sorts these queues and then merges the sorted queues into the original queue.

    Write a recurrence formula for the time complexity of this variant and show the order of magnitude of the time complexity as a function of n, where n is the number of values in the original queue.

    Svar/Answer:
    Rakningin er $T(n) = 5T(\frac{n}{5}) + O(n)$, stærðargráðan er $T(n) = O(n \log n)$.

12. Hverjar eru stærðargráður fallanna sem hafa eftirfarandi rakningarvensl?

    What are the orders of magnitude of the functions that have the following recurrence relations?

(a) $T(n) = 25T(n/5) + O(n)$

(b) $T(n) = 3T(n/3) + O(n)$

(c) $T(n) = 81T(n/3) + O(n)$

(d) $T(n) = 2T(n/3) + O(n)$

Svar/Answer:

(a) $O(n^2)$

(b) $O(n \log n)$

(c) $O(n^4)$

(d) $O(n)$