Istanbul Bilgi University
Department of Computer Science
Comp 231
Binary Exam Question Pool

Laszlo Nemeth Can Burak Çilingir Coşkun Gündüz

November 28, 2006

Contents

1 Orientation

This document collects projects which should be useful to understand the material in COMP 231, and improve your programming skills. All the projects require you to

1. Implement it in Java,
2. Work out a few (at least three) examples by hand,
3. Show that your program produces the expected output.

The functions you write must throw exceptions generated by you for invalid or irrelevant inputs or for cases where an a result is not possible. For example if you are searching a number and can't find it, instead of returning -1, you must throw an exception.

Some descriptions are from Dictionary of Algorithms and Data Structures[1] and some are from Wolfram MathWorld[2].

Note that only questions are labelled as following will be asked in the binary exam.

*** (3 stars)

**** (4 stars)

***** (5 stars)

2 Questions

2.1 Single Dimensional Matrices

2.1.1 Binary Search

This question is rated as: **

Search a sorted array by repeatedly dividing the search interval in half. Begin with an interval covering the whole array. If the value of the search key is less than the item in the middle of the interval, narrow the interval to the lower half. Otherwise narrow it to the upper half. Repeatedly check until the value is found or the interval is empty.

```
int binarySearch(int[] array, int number)
```

Special cases If array is of odd length, slice at floor(arr.length/2) If there are more than one of the same element, return the index of the first occurence.

Parameters A sorted array of integers and an integer to be searched in this array.

Return value Index of the number in the array.

Examples `int[] a = {-2,0,3,5,7,7,9,12}`

```
binarySearch(a, 3) is 2
binarySearch(a, 7) is 4
```

[1] http://www.nist.gov/dads/
[2] http://mathworld.wolfram.com/

2.1.2 Fibonaccian Search

*This question is rated as: ****

Search a sorted array by narrowing possible locations to progressively smaller intervals. Begin with two Fibonacci numbers, $p(F(n))$ and $q(F(n+1))$, such that $p < n \leq q$, where n is the size of the array. The first step checks location p. The size of the next interval is p, if the key is less than the item at that location, or $q - p(F(n-1))$ if it is greater.

Note: This is similar to a binary search, but only needs subtraction, instead of divide by two or shift right, to compute the next position.

```
int fibonacciSearch(int[] array, int number)
```

Special cases If there are more than one of the same element, return the index of the first occurence.

Parameters A sorted array of integers and an integer to be searched in this array.

Return value Index of the number in the array.

Examples `int[] a = {-2,0,3,5,7,7,9,12}`

```
fibonacciSearch(a, 3) is 2
fibonacciSearch(a, 7) is 4
```

2.1.3 Gaussian Elimination

*This question is rated as: ****

```
double[][] gaussianElimination(double [][])
```

Parameters A 2D double array.

Return value A 2D double array which is upper triangular. An upper triangular matrix is a matrix that is only defined at (i, j) when $i \leq j$.

2.1.4 Hamming Distance

*This question is rated as: **

```
int hammingDistance(String a, String b)
```

Parameters Two binary strings with same size consist of 0's and 1's those represent numbers in base 2.

Return value An integer that is the Hamming distance between two input strings. If there are elements in arrays which are not 0 or 1, than an exception must be thrown for invalid input.

2.1.5 Matrix mode

*This question is rated as: *****

```
int[] mode (int[] array)
```

Parameters An array of integers, not necessarily sorted.

Return value An array containing the value which occurs most often. If no value is repeated, there is no mode. If more than one value occurs with the same greatest frequency, each value should be returned in the array..

Examples `int[] a = {4,5,7,3,2,5,6,8}`

```
mode(a) is {5}

int[] b = {4,6,3,4,7,8,2,7}

mode(b) is {4,7}
```

2.1.6 Select and partition

*This question is rated as: ****

Given an array A of n elements and a positive integer $k \leq n$, partition the array such that

$$A[1], \ldots, A[k-1] \leq A[k] \leq A[k+1], \ldots, A[n]$$

```
int[] partition(int[] array, int k)
```

Parameters An array of integers, integer.

Return value An integer array

Examples `int[] a = {2,6,1,7,3,4,6}`

```
partition(a,5) is, say {2,1,3,4,6,6,7}
```

2.1.7 Subarray

*This question is rated as: ***

An array v occurs in another array u if v is a subarray of u.

```
boolean doesOccur(int[] haystack, int[] needle)
```

Parameters 2 arrays of integers.

Return value True if second array is contained in first array. False otherwise.

Examples `int[] a = {4,5,1,6,7,4}`
`int[] b = {1,6,7}`

```
doesOccur(a, b) is true
doesOccur(b, a) is false
```

2.1.8 Decide uniqueness

*This question is rated as: ***

The problem of determining if there are duplicates in a list of numbers.

```
boolean areAllElementsAreUnique(int[] array)
```

Parameters An array of integers.

Return value True, if all the numbers are occuring exacly one time in the given array. Otherwise false.

Examples `int[] a = {2,6,1,7,3,4,6}`

```
areAllElementsAreUnique(a) is false

int[] b = {1,6,3,2,9,5}

areAllElementsAreUnique(b) is true
```

2.2 Two Dimensional Matrices

2.2.1 Matrix addition

This question is rated as: **

Add 2 matrices.

```
int[][] add(int[][] a, int[][] b)
```

Parameters Two 2D arrays of integers.

Return value Sum of the two matrices without changing the contents of the parameters (non-destructive).

2.2.2 Matrix determinant

This question is rated as: *****

Calculate the detarminant of a matrix. Text below is from Wolfram MathWorld[3]. See the URL for more information.
A general determinant for a matrix A has a value

$$|A| = \sum_{i=1}^{k} a_{ij} C_{ij}$$

with no implied summation over j and where C_{ij} (also denoted a^{ij}) is the cofactor of a_{ij} defined by

$$C_{ij} = (-1)^{i+j} M_{ij}$$

and M_{ij} is the minor of matrix A formed by eliminating row i and column j from A. If $i = 2$ and $j = 2$, the minor (M_{22}) of the matrix

$$A = \begin{bmatrix} & \Downarrow & & & \\ & a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ \Rightarrow & a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ & a_{31} & a_{32} & a_{33} & \dots & a_{3n} \\ & \vdots & \vdots & \vdots & \ddots & \vdots \\ & a_{k1} & a_{k2} & a_{k3} & \dots & a_{kn} \end{bmatrix}$$

is

$$M_{22}(A) = \begin{bmatrix} a_{11} & a_{13} & \dots & a_{1n} \\ a_{31} & a_{33} & \dots & a_{3n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{k1} & a_{k3} & \dots & a_{kn} \end{bmatrix}$$

```
int determinant(int[][] a)
```

Parameters 2D array of integers.

Return value Determinant of the matrix.

[3] http://mathworld.wolfram.com/Determinant.html

2.2.3 Matrix inversion

This question is rated as: ******

This question will not occur in the binary exam.
Calculate the inverse of a matrix.

```
int[][] invert(int[][] a)
```

Parameters 2D array of integers.

Return value Inverse of the matrix without changing the contents of the parameter (non-destructive).

2.2.4 Matrix multiplication

This question is rated as: ***

Multiply 2 matrices.

```
int[][] multiply(int[][] a, int[][] b)
```

Parameters Two 2D arrays of integers.

Return value Product of the two matrices without changing the contents of the parameters (non-destructive).

2.2.5 Matrix transpose

This question is rated as: **

Calculate the transposition of a matrix.

```
int[][] transpose(int[][] a)
```

Parameters 2D array of integers.

Return value Transposition of the matrix without changing the contents of the parameter (non-destructive).

2.3 Binary Numbers

2.3.1 Binary Addition

This question is rated as: ***

```
int[] addBinary(int[] a, int []b)
```

Parameters Two integer arrays consist of 0's and 1's those represents numbers in base 2.

Return value Another integer array which consists of 0's and 1's that is the sum of two binary numbers passed as parameters. If there are elements in arrays which are not 0 or 1, than an exception must be thrown for invalid input. The size of two input arrays may be different.

Examples
```
int[] a={1,0,1,1,0}
int[] b={1,0,1}

The resulting array should be: {1,1,0,1,1}
```

2.3.2 Binary Multiplication

*This question is rated as: *****

```
int[] multiplyBinary(int[] a, int []b)
```

Parameters Two integer arrays consist of 0's and 1's those represents numbers in base 2.

Return value Another integer array which consists of 0's and 1's that is the multpilication of two binary numbers passed as parameters. If there are elements in arrays which are not 0 or 1, than an exception must be thrown for invalid input. The size of two input arrays may be different.

Examples `int []a={1,0,1,1,0}`
`int []b={1,0,1}`

`The resulting array should be: \{1,1,0,1,1,1,0\}`

2.3.3 Binary Subtraction

*This question is rated as: *****

```
int [] subtractBinary(int[] a, int []b)
```

Parameters Two integer arrays consist of 0's and 1's those represents numbers in base 2.

Return value Another integer array which consists of 0's and 1's that is the subtraction of the second parameter from the first one. i.e. $a - b$ If there are elements in arrays which are not 0 or 1, than an exception must be thrown for invalid input. If the result of the subtraction is negative than an exception must be thrown. The size of two input arrays may be different.

Examples `int []a={1,0,1,1,0}`
`int []b={1,0,1}`

`The resulting array should be: {1,0,0,0,1}`

2.4 Lists

Here is a specification of a few functions over lists.

	nil	: List a
	cons	: a →List a →List a
*	null	: List a →Bool
	null nil	= true
	null (cons x xs)	= false
*	head	: List a →a
	head nil	= error "head"
	head (cons x xs)	= x
*	elem	: a →List a →Bool
	elem x nil	= false
	elem x (cons y ys)	= **if** (x == y) **then** true **else** elem x ys
*	length	: List a →Int
	length nil	= 0
	length (cons x xs)	= 1 + length xs
**	delFirst	: a →List a →List a
	delFirst x nil	= nil
	delFirst x (cons y ys)	= **if** (x == y) **then** ys **else** cons y (delFirst x ys)
**	delAll	: a →List a →List a
	delAll x nil	= nil
	delAll x (cons y ys)	= **if** (x == y) **then** delAll x ys **else** cons y (delFirst x ys)
**	before	: a →List a →List a
	before x nil	= nil
	before x (cons y ys)	= **if** (x == y) **then** nil **else** cons y (before x ys)
**	append	: List a →List a →List a
	append nil ys	= xs
	append (cons x xs) ys	= cons x (append xs ys)
**	reverse	: List a →List a
	reverse nil	= nil
	reverse (cons x xs)	= append (reverse xs) (cons x nil)
**	take	: Int →List a →List a
	take 0 xs	= nil
	take n nil	= nil
	take n (cons x xs)	= cons x (take (n-1) xs)

Implement 2 of "**" and 1 of "*" functions for the following lists in this section and run each function of your code on random data for

$$10, 50, 100, 500, 1000, 5000, 10000, 50000, 100000$$

inputs and measure the runtimes. Plot the measured times against the number of inputs on a graph paper.

Note that all operations replace the elements of the list in place.

2.4.1 singly-linked, sorted, circular

*This question is rated as: ******

Implement the functions and run tests on your code as described in the beginning of this section.

2.4.2 singly-linked, sorted, not-circular

*This question is rated as: ******

Implement the functions and run tests on your code as described in the beginning of this section.

2.4.3 singly-linked, unsorted, circular

*This question is rated as: ******

Implement the functions and run tests on your code as described in the beginning of this section.

2.4.4 singly-linked, unsorted, not-circular

*This question is rated as: ******

Implement the functions and run tests on your code as described in the beginning of this section.

2.4.5 doubly-linked, sorted, circular

*This question is rated as: ******

Implement the functions and run tests on your code as described in the beginning of this section.

2.4.6 doubly-linked, sorted, not-circular

*This question is rated as: ******

Implement the functions and run tests on your code as described in the beginning of this section.

2.4.7 doubly-linked, unsorted, circular

*This question is rated as: ******

Implement the functions and run tests on your code as described in the beginning of this section.

2.4.8 doubly-linked, unsorted, not-circular

*This question is rated as: ******

Implement the functions and run tests on your code as described in the beginning of this section.

2.5 Number Questions

2.5.1 Cube root

*This question is rated as: ***

```
double cbrt(double a)
```

Parameters A real number.

Return value $\sqrt[3]{a}$

2.5.2 Greatest Common Divisor

*This question is rated as: ***

```
int gcd(int, int)
```

Parameters Two integers x and y.

Return value Another integer, the greatest common divisor of two input integers. An exception must be thrown for invalid input.

2.5.3 Least Common Multiplier

*This question is rated as: ***

```
int lcm(int, int)
```

Parameters Two integers x and y.

Return value Another integer, the least common multiplier of two input integers. An exception must be thrown for invalid input.

2.5.4 Repeated Squaring

*This question is rated as: ****

```
int repeatedSquare(int, int)
```

Parameters Two integers x and n such that x^n will be calculated.

Return value Another integer. An exception must be thrown for invalid input.

2.6 Sorting Questions

2.6.1 Bubble Sort

*This question is rated as: **

```
int [] bubbleSort(int[])
```

Parameters An array of integers

Return value An array sorted by comparing each adjacent pair of items in an array in turn, swapping the items if necessary, and repeating the pass through the array until no swaps are done.

Test your code for multiple input size.

2.6.2 Heap Sort

*This question is rated as: ******

```
int [] heapSort(int[])
```

Parameters An array of integers

Return value An array sorted by building a heap, then repeatedly extracting the maximum item.

Test your code for multiple input size.

2.6.3 Insertion Sort

*This question is rated as: **

```
int [] insertionSort(int[])
```

Parameters An array of integers

Return value An array sorted by repeatedly taking the next item and inserting it into the final data structure in its proper order with respect to items already inserted.

Test your code for multiple input size.

2.6.4 Quick Sort

*This question is rated as: ****

`int [] quickSort(int[])`

Parameters An array of integers

Return value An array sorted by picking an element from the array (the pivot), partition the remaining elements into those greater than and less than this pivot, and recursively sort the partitions. About choosing the pivot element use the three methods below.

1. Choose the element with index $(n+1)/2$
2. Pick the pivot of three fixed position like first, middle and end.
3. Randomly choose three indexes and pick the median of these elements as pivot.

Test your code for multiple input size.

2.6.5 Radix Sort

*This question is rated as: *****

`int [] radixSort(int[])`

Parameters An array of integers

Return value An array sorted by a multiple pass distribution sort algorithm that distributes each item to a bucket according to part of the item's key beginning with the least significant part of the key. After each pass, items are collected from the buckets, keeping the items in order, then redistributed according to the next most significant part of the key.

Test your code for multiple input size.

2.6.6 Selection Sort

*This question is rated as: **

`int [] selectionSort(int[])`

Parameters An array of integers

Return value An array sorted by repeatedly looking through remaining items to find the least one and moving it to its final location.

Test your code for multiple input size.

2.6.7 Shell Sort

*This question is rated as: ****

`int [] shellSort(int[])`

Parameters An array of integers

Return value The first diminishing increment sort. On each pass i sets of n/i items are sorted, typically with insertion sort. On each succeeding pass, i is reduced until it is 1 for the last pass. A good series of i values is important to efficiency.

Test your code for multiple input size.

2.7 Trees

2.7.1 Left threading tree

*This question is rated as: ***

Parameters a tree

Return value a left threaded tree Write a function which takes a binary tree and converts it to a left-threaded tree. For a detailed discussion of threaded trees see Knuth: The Art Of Computer Programming (Fundamental Algorithms - Vol I) 2.3.1.

2.7.2 Unique tree from postorder and inorder reversal

*This question is rated as: **

Parameters The postorder and the inorder traversal of a tree

Return value A tree

Suppose you are given the postorder $\mathcal{O}$ and the inorder $\mathcal{I}$ of the nodes of a binary tree in which all nodes are unique (every key occurs at most once). Design and implement an algorithm which constructs the tree, whose postorder and inorder are $\mathcal{O}$ and $\mathcal{I}$ respectively.

2.7.3 Unique tree from preorder and inorder reversal

*This question is rated as: **

Parameters The preorder and the inorder traversal of a tree

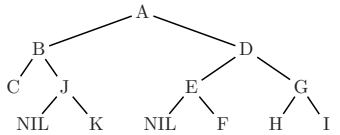
Return value A tree

Suppose you are given the preorder $\mathcal{E}$ and the inorder $\mathcal{I}$ of the nodes of a binary tree in which all nodes are unique (every key occurs at most once). Design and implement an algorithm which constructs the tree, whose preorder and inorder are $\mathcal{E}$ and $\mathcal{I}$ respectively.

For example, if

$$\begin{aligned} \mathcal{I} &= C\ B\ J\ K\ A\ E\ F\ D\ H\ G\ I \\ \mathcal{E} &= A\ B\ C\ J\ K\ D\ E\ F\ G\ H\ I \end{aligned}$$

then there is exactly one tree which can be constructed (note: NIL is shown only to force K to be the right subtree of J):



2.7.4 Right threading a tree

*This question is rated as: ***

Parameters a tree

Return value a right threaded tree

Write a function which takes a binary tree and converts it to a right-threaded tree. For a detailed discussion of threaded trees see Knuth: The Art of Computer Programming (Fundamental Algorithms - Vol I) 2.3.1.

2.7.5 Threading a tree

*This question is rated as: ****

Parameters a tree

Return value a threaded tree

Write a function which takes a binary tree and converts it to a threaded tree (with left and right threads). For a detailed discussion of threaded trees see Knuth: The Art of Computer Programming (Fundamental Algorithms - Vol I) 2.3.1.

2.7.6 Deletion from an AVL tree

*This question is rated as: ****

Parameters An element x of some type (for example Int), and a red-black tree

Return value an AVL tree with the element deleted

2.7.7 Insertion into an AVL tree

*This question is rated as: ****

Parameters An element x of some type (for example Int), and a red-black tree

Return value an AVL tree with the element x inserted

2.7.8 Deletion from a binary search tree

*This question is rated as: ****

Parameters An element x of some type (for example Int), and a binary search tree

Return value a binary search tree with the element deleted There are several variations on this question: if you allow duplicates in the tree, you have to decide if your deletion should delete one occurrence or all occurrences.

2.7.9 Insertion into a binary search tree

*This question is rated as: ****

Parameters An element x of some type (for example Int), and a binary search tree

Return value a binary search tree with the element inserted There are several variations on this question. You may want to consider allowing duplicates in the tree.

2.7.10 Inorder traversal

*This question is rated as: ***

Parameters Binary tree

Return value A list containing containing all the items of the tree in 'inorder'.

Notice, that the traversal is independent of what you do at the nodes: you can generalise this such that the traversal function takes another function f as an argument and applies f to all the nodes of the tree.

2.7.11 Postorder traversal

*This question is rated as: ****

Parameters n-ary tree

Return value A list containing containing all the items of the tree in 'postorder'.

Notice, that the traversal is independent of what you do at the nodes: you can generalise this such that the traversal function takes another function f as an argument and applies f to all the nodes of the tree.

2.7.12 Preorder traversal

*This question is rated as: ****

Parameters n-ary tree

Return value A list containing containing all the items of the tree in 'preorder'.

Notice, that the traversal is independent of what you do at the nodes: you can generalise this such that the traversal function takes another function f as an argument and applies f to all the nodes of the tree.

2.7.13 Postorder traversal

*This question is rated as: ***

Parameters Binary tree

Return value A list containing containing all the items of the tree in 'postorder'.

Notice, that the traversal is independent of what you do at the nodes: you can generalise this such that the traversal function takes another function f as an argument and applies f to all the nodes of the tree.

2.7.14 Preorder traversal

*This question is rated as: ***

Parameters Binary tree

Return value A list containing containing all the items of the tree in 'preorder'.

Notice, that the traversal is independent of what you do at the nodes: you can generalise this such that the traversal function takes another function f as an argument and applies f to all the nodes of the tree.

2.7.15 Deletion from a red-black tree

*This question is rated as: ****

Parameters An element x of some type (for example Int), and a red-black tree

Return value a red-black tree with the element deleted

2.7.16 Insertion into a red-black tree

*This question is rated as: ****

Parameters An element x of some type (for example Int), and a red-black tree

Return value a red-black tree with the element x inserted