

1. Matching [7 pts]

Select the **best** word for each of the definitions below by writing the appropriate letter in the blank beside the word.

1. _____ An output stream for reporting error messages.
2. _____ A construct used to specify how to deal with exceptions.
3. _____ An object that is thrown in the event of an error that the method cannot handle.
4. _____ A generic term for something from which data can be read or written
5. _____ The percentage of full spaces in a HashTable.
6. _____ An interface that specifies hasMoreElements() and nextElement().
7. _____ A recursively defined data structure in which each node can have up to two children.

- A. Stream
- B. Decorator Class
- C. Standard Output
- D. Quadratic Probing
- E. Enumeration
- F. Collision
- G. Exception
- H. Standard Error
- I. Try-catch
- J. Finally
- K. External Chaining
- L. Linear Probing
- M. Load Factor
- N. Binary Search Tree
- O. String Tokenizer
- P. Throw
- Q. Rehash
- R. Standard Input
- S. Binary Tree

2. Swap [10 pts]

A java programmer wrote the following code:

```
public class Swapper{
    public static void swap(int a, int b){
        int temp=a;
        a=b;
        b=a;
    }//swap
    public static void main(String[] args){
        int x=3;
        int y=57;
        swap(x,y);
        System.out.println("x = " + x +" y = " + y);
    }//main
}//class Swapper
```

The java programmer expected the output to be

x = 57 y = 3

- 1 (a) Is this the actual output? If not, what is the correct output?
- 4 (b) If the program did not work as the programmer expected, explain why. If it did work as expected, explain why it works.

Another java programmer wrote the following code:

```
public class ArraySwapper{
    public static void swap(int[] data, int index1, int index2){
        int temp=data[index1];
        data[index1]=data[index2];
        data[index2]=temp;
    }//swap
    public static void main(String[] args){
        int[] data={9,33};
        swap(data,0,1);
        System.out.println("data[0] = "+data[0]+" data[1] = "+data[1]);
    }//main
}//end class ArraySwapper
```

The java programmer expected the output to be

data[0] = 33 data[1] = 9

- 1 (c) Is this the actual output? If not, what is the correct output?
- 4 (d) If the program did not work as the programmer expected, explain why. If it did work as expected, explain why it works.

3. Exceptions [10 pts]

Given the following code fragment (with some lines numbered on the left- note that the line numbers do not affect the code's functionality):

```
    try{
1:        x=3;
2:        somethingHere();
3:        y=7;
    }
    catch(IOException ioe) {
4:        System.err.println(ioe);
    }
    catch(NumberFormatException nfe) {
5:        System.err.println(nfe);
    }
    catch(Exception e) {
6:        System.err.println(e);
    }
    finally {
7:        System.out.println("all done");
    }
8: System.out.println("the end.");
```

Write **in the correct order** the *line numbers* for the lines that execute in the code above if

5 (a) **somethingHere()** does not throw any type of Exception.

5 (b) **somethingHere()** throws a `NumberFormatException`.

4. File IO [16 pts]

Write the method `public void writeObjectToFile(Object o, String filename)` which

1. opens `filename` appropriately to write Objects
2. writes `o` to it
3. ensures that everything is properly closed
4. if any errors occur, the message `An error occurred` is printed to standard output

You may wish to remember the following classes from P3:

- `FileOutputStream`
- `ObjectOutputStream`

```
public void writeObjectToFile(Object o, String filename) {
```

```
}
```

5. HashTables [15 pts]

- 7 (a) Show the resulting HashTable when the following <key,data> pairs are added. The un-modded hashCode for the key is given in parenthesis after the pair. (You may add to the front or to the back, whichever you prefer).
- <"Pluto", "Hades"> (52)
 - <"Mars", "Ares"> (26)
 - <"Mercury", "Hermes"> (12)
 - <"Venus", "Aphrodite"> (17)
 - <"Jupiter", "Zeus"> (4)
 - <"Neptune", "Poseidon"> (73)
 - <"Saturn", "Cronus"> (14)

The HashTable is of size 5 and uses external chaining. (Hint: you will be drawing a picture to answer this question). You may show only the **data** in your drawing.

- 4 (b) Explain what a collision is, and why they are a problem.

- 4 (c) Besides external chaining, name and explain another collision resolution method used in HashTables.

6. Useful Tools [10 (+5) pts]

For this question you may choose to do **either** (a) **or** (b) for the full 10 points, or you may do **both** for **+5 extra credit**. If you do both, it will be assumed that whichever you write first is to be graded for 10 points, and whichever is second is to be graded for 5 points of extra credit, unless you indicate otherwise. You may wish to recall that Enumeration is declared as

```
public interface Enumeration{
    public boolean hasMoreElements();
    public Object nextElement();
} //end Enumeration
```

- (a) Write the method **public void printEnum(Enumeration e)** which prints out all elements in the Enumeration **e**, one per line.
- (b) Write the method **public String lastWord(String s)** which returns the last word in the String. Words are delimited by whitespace (space, tab, or newline)- the default delimiter for **StringTokenizer**. If **s** is the empty String, this method should return the empty String, **""**. You may assume that **s** is not *null*.

7. HashTables [16 pts]

Give the following HashNode class:

```
public class HashNode{
    private Object data;
    private Object key;
    private HashNode next;
    public HashNode(Object k, Object d){key=k; data=d;}
    public void setData(Object d){data=d;}
    public Object getData() {return data;}
    public void setKey(Object k){key=k;}
    public Object getKey() {return key;}
    public void setNext(HashNode n) {next=n;}
    public HashNode getNext() {return next;}
} //class HashNode
```

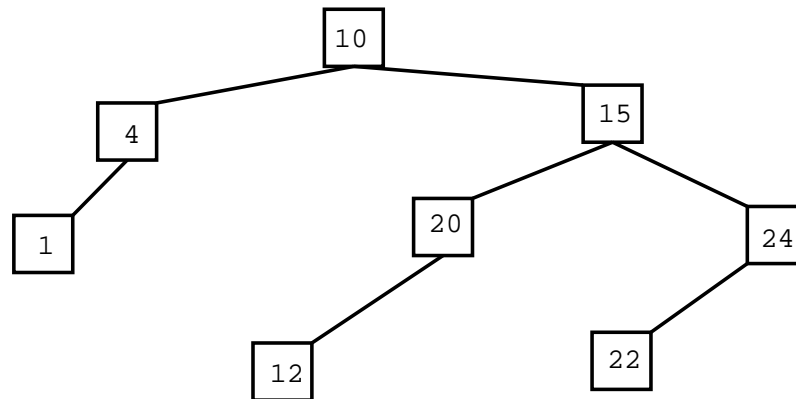
Write the method **public void add(Object key, Object data)** in class **HashTable**. You may write any helper methods you wish. The HashTable is externally chained- you may add to the front or the back of the chain, whichever you prefer.

```
public class HashTable {
    private HashNode [] hTable;
    public HashTable(int size) {
        hTable=new HashNode[size];
    } //HashTable(int)
    public int getHashKey(Object key) {
        return Math.abs(key.hashCode() % hTable.length);
    }
    public void add(Object key, Object data){
        // YOUR CODE GOES HERE

}
}
```

8. Binary Search Trees [16 pts]

Given the following BST:



- 4 (a) Add 0, 5, 13, 21 (in that order) to the tree, and draw the resulting tree. You only need to draw the final tree. Draw your changes on the tree above.
- 4 (b) Perform an inorder traversal of the **original** tree, printing each node, and write the output:
- 4 (c) Perform an preorder traversal of the **original** tree, printing each node, and write the output:
- 4 (d) Perform an postorder traversal of the **original** tree, printing each node, and write the output: