

AP COMPUTER SCIENCE A – PrintWriter CONNECTED TO A FILE

Min Sort Algorithm (also called a Selection Sort)

Say we have a list of numbers, and we want to sort the numbers into ascending order. The method `sortList` in file `MinSortStub.java` performs the sort with the min sort algorithm. Note that `int [] intA` is a private field of the class.

```

/**/
public void sortList() {
    /**/
    int numInt=intA.length;
    int min, minJ;
    for ( int iPass=0; iPass<numInt-1; ++iPass ) {
        min=Integer.MAX_VALUE;
        minJ=-1;
        for ( int j=iPass; j<numInt; ++j ) {
            if ( intA[j] < min ) {
                min=intA[j];
                minJ=j;
            }
        }
        switchInt (minJ, iPass);
    }
    /**/
    return;
}
/**/
private void switchInt ( int i, int j ){
    /**/
    int p1=intA[i];
    int p2=intA[j];
    intA[i]=p2;
    intA[j]=p1;
    /**/
    return;
}

```

Say that the list we are sorting has six entries as shown in the leftmost column of the table. After each execution of the `ipass` loop, the list is shown in the table. Note that for six entries, five passes are executed.

after ipass =

	0	1	2	3	4
4	0	0	0	0	0
2	2	1	1	1	1
5	5	5	2	2	2
1	1	2	5	3	3
0	4	4	4	4	4
3	3	3	3	5	5

Formatting Output

```
public class FormatTest {  
    /**/  
    public static void main ( String [] arg ) {  
        /**/  
        int i;  
        double d;  
        String iS, dS;  
        /**/  
        i=-689;  
        d=38.479;  
        /*  
        *   iS = a decimal integer right-justified  
        *       in a field-width of 6.  
        */  
        iS=String.format("%6d",i);  
        /*  
        *   dS = a floating-point number right-justified  
        *       in a field-width of 5, with 2 digits after  
        *       the decimal.  
        */  
        dS=String.format("%5.2f",d);  
        /**/  
        System.out.println( "\"" + iS + "\"");  
        System.out.println( "\"" + dS + "\"");  
        /**/  
        return;  
    }  
}
```

Running program FormatTest gives the output

```
"  -689"  
"38.48"
```

AP COMPUTER SCIENCE A – PrintWriter CONNECTED TO A FILE

Program MinSort

The file `MinSortStub.java` contains the same methods as did program `MaxSumTest`, i.e., `loadFileAsIntArray`, `openBR`, `readLine` and `close`, which methods read the file `intList.txt` into the program. `MinSortStub` also contains the methods which perform the min sort. Finally, it has the method `openPW`, which opens a `PrintWriter` object connected to an output file

```

/*
 * Open a PrintWriter connected to fileName.
 */
private static PrintWriter openPW ( String fileName ) {
    /**/
    PrintWriter rv;
    /**/
    rv=null;
    try {
        rv=new PrintWriter(fileName);
    }
    catch ( FileNotFoundException fnfe ) {
        System.out.println();
        System.out.println( "Could not open \"" + fileName + "\" for output." );
        System.exit(0);
    }
    /**/
    return rv;
}

```

must catch, not a RuntimeException

Will only know if filename represents a file in a directory (or folder) to which you do not have access.

and the method `printSort`, which prints the sorted list to the output file.

```

/**/
public void printSort ( String outputFile ) {
    /**/
    PrintWriter pw;
    /**/
    pw=openPW(outputFile);
    for ( int i=0; i<intA.length; ++i ) {
        pw.println( String.format("%2d",intA[i]) );
    }
    pw.close();
    /**/
    return;
}

```

d ≡ decimal integer

"%2d" put intA[i] right-justified in a field-width of 2

4 OF 4

Finish the main method of file `MinSortStub` to make a program `MinSort`.

- 1) Check that the program is invoked with two arguments, *i.e.*, the input file name and output file name. If it is not, print a usage statement to the console such as

Usage:

```
java MinSort inputFile outputFile
```

and return.

- 2) Next instantiate a new `MinSort(arg[0])`, call `sortList()`, and then call `printSort(arg[1])`.

- 3) Run the program using `intList.txt` as the input file. Your output file should look like

```
7
8
16
17
19
25
27
28
29
32
37
39
42
43
59
71
73
78
83
93
```