1.124J Foundations of Software Engineering

Problem Set 6

Due Date: Thursday 11/2/00

Reference Readings: *From Java Tutorial*

- *All material from the previous problem set*
- *Essential Java Classes:*
  - *Using String and StringBuffer (recommended)*
  - *Setting Program Attributes*
  - *Accessing System Resources*
  - *Handling Errors with Exceptions (required)*
  - *Doing Two or More Tasks At Once: Threads (required)*
  - *Reading and Writing (required)*
- *Writing Applets*
  - *Overview of Applets (required)*
  - *Taking Advantage of the Applet API (recommended)*
  - *Practical Considerations when Writing Applets (recommended)*
  - *Finishing an Applet (recommended)*
- *Creating a GUI with JFC/Swing*
  - *Getting Started with Swing (required)*
  - *Swing Features and Concepts (required)*
  - *Using Swing Components (required)*
  - *Using Other Swing Features (required)*
  - *Laying Out Components Within a Container*
  - *Writing Event Listeners (required)*
  - *Working with Graphics*
  - *Converting to Swing*
- *Java2D*
  - *Overview of the Java 2D API*
  - *Displaying Graphics with Graphics2D*
    - *Stroking and Filling Graphics Primitives*

Problem 1: [60%]

In this problem you need to write a Java application to read the x and y coordinates from a file, named ***data6_1*** and make a plot of the values. It is advised to implement the functionality in the introductory part and then start building the program gradually.

Introductory Part:

An issue that you have to address is the data input from a file. It may be easier if you write the class to open a file and read the values stored in that file and then proceed to the graphical user interface part.

You need to write a Java application to read from the file *data6_1* the x and y coordinates of points. The file has first the number of points (i.e. couples of x and y values) stored.

Your application should be able to open the file, read the values into arrays of double, named x and y, and print out their values. Write a class with a method to achieve this functionality.

>java Sol6_1

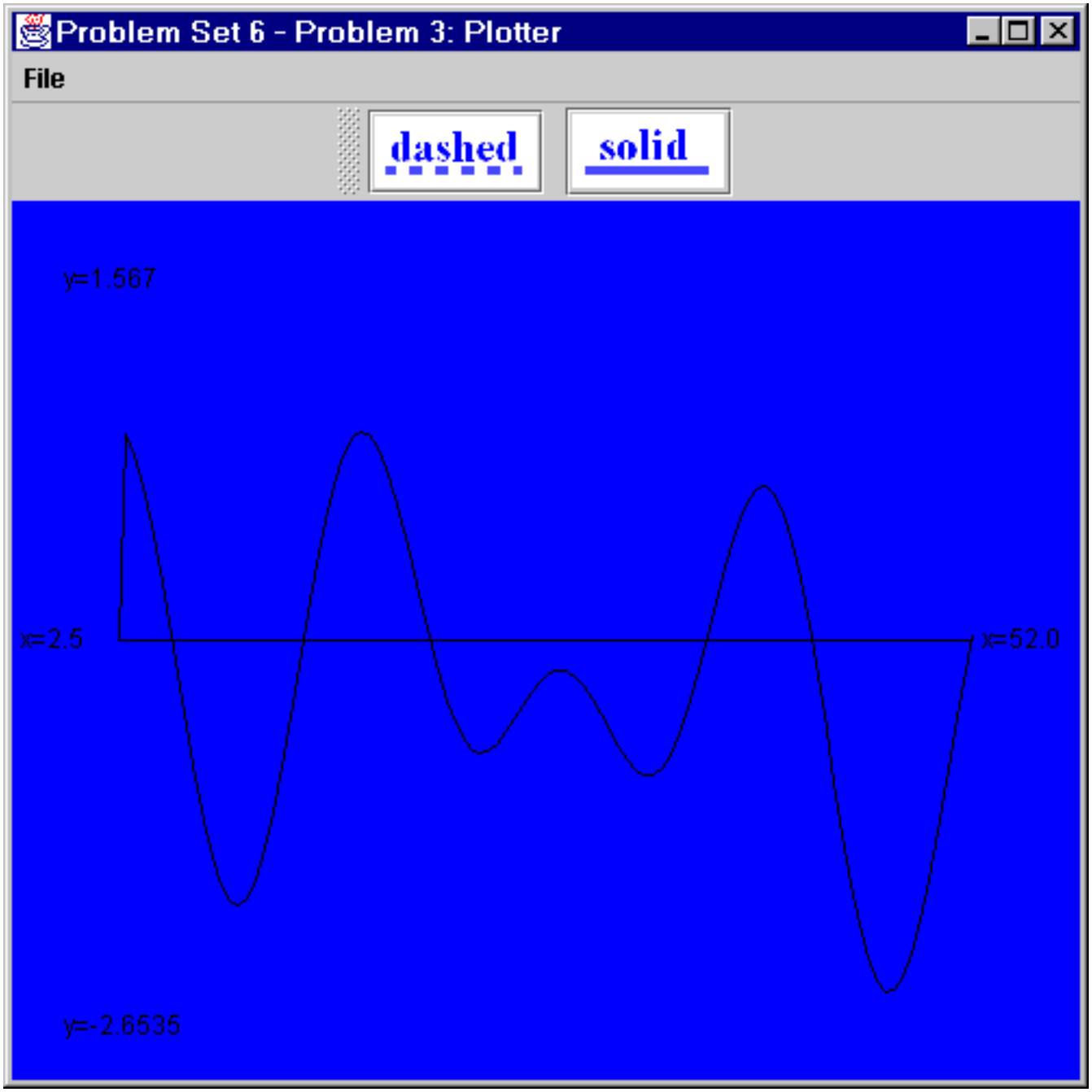
100 points have been read

x[1]=2.5 y[1]=2.5
x[2]=3.0 y[2]=3.0
x[3]=3.5 y[3]=3.5
.....

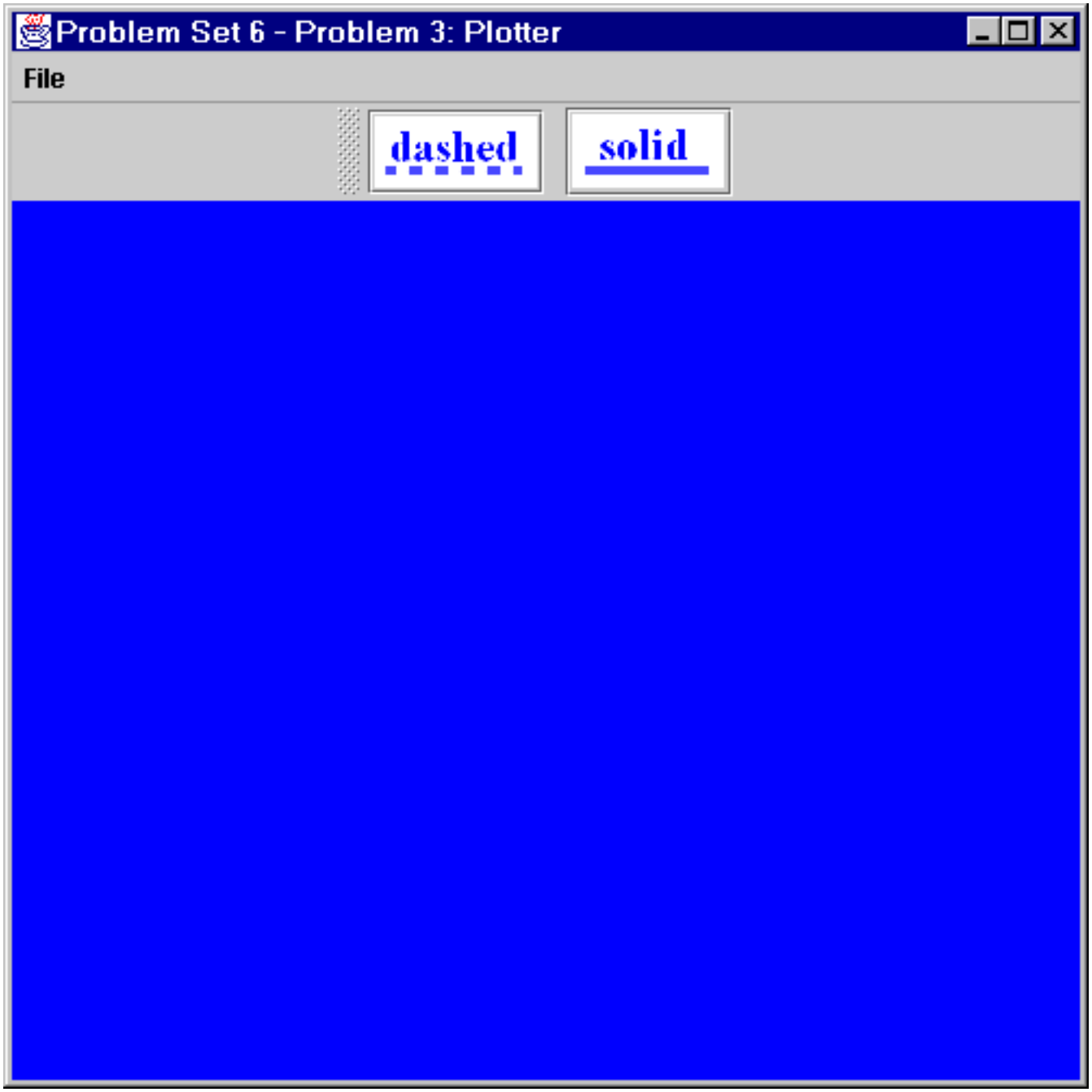
Main part:

Having implemented the part to read values from a file you should proceed to the design of the graphical user interface. This should be done very gradually to avoid errors and needless waste of time during debugging. You should write part by part, e.g. just a *JFrame*, then add a *JMenuBar*, then add a *JToolBar*, etc., checking each part before proceeding to the next one.

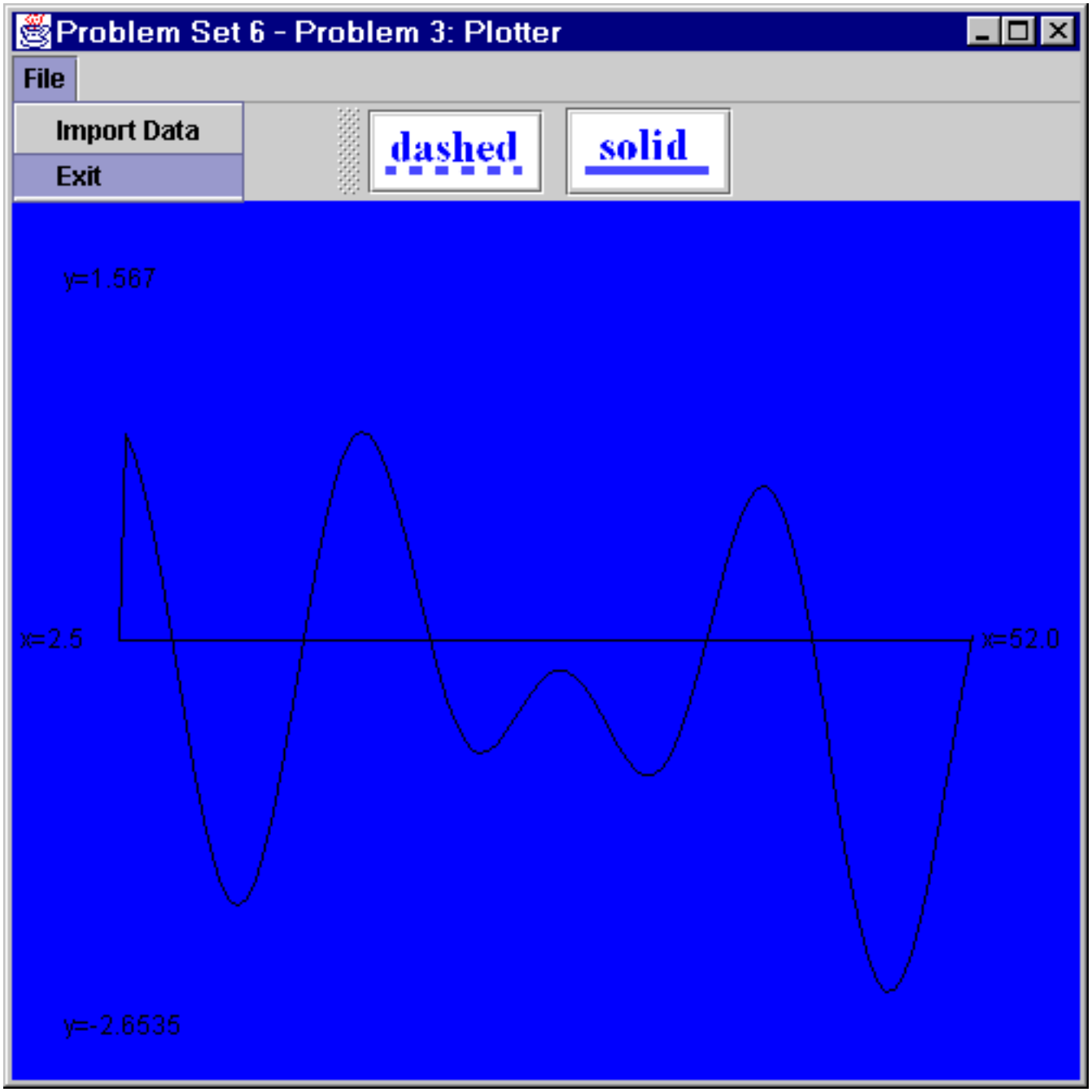
Ultimately, you need to write a Java application that will read in values of points from a file and plot a polyline joining those points as shown below, after reading in the values:



Initially, the program should have the following look:



It should have a menu bar with a menu named File. Under the "File" menu there should be two menu items, as shown below:



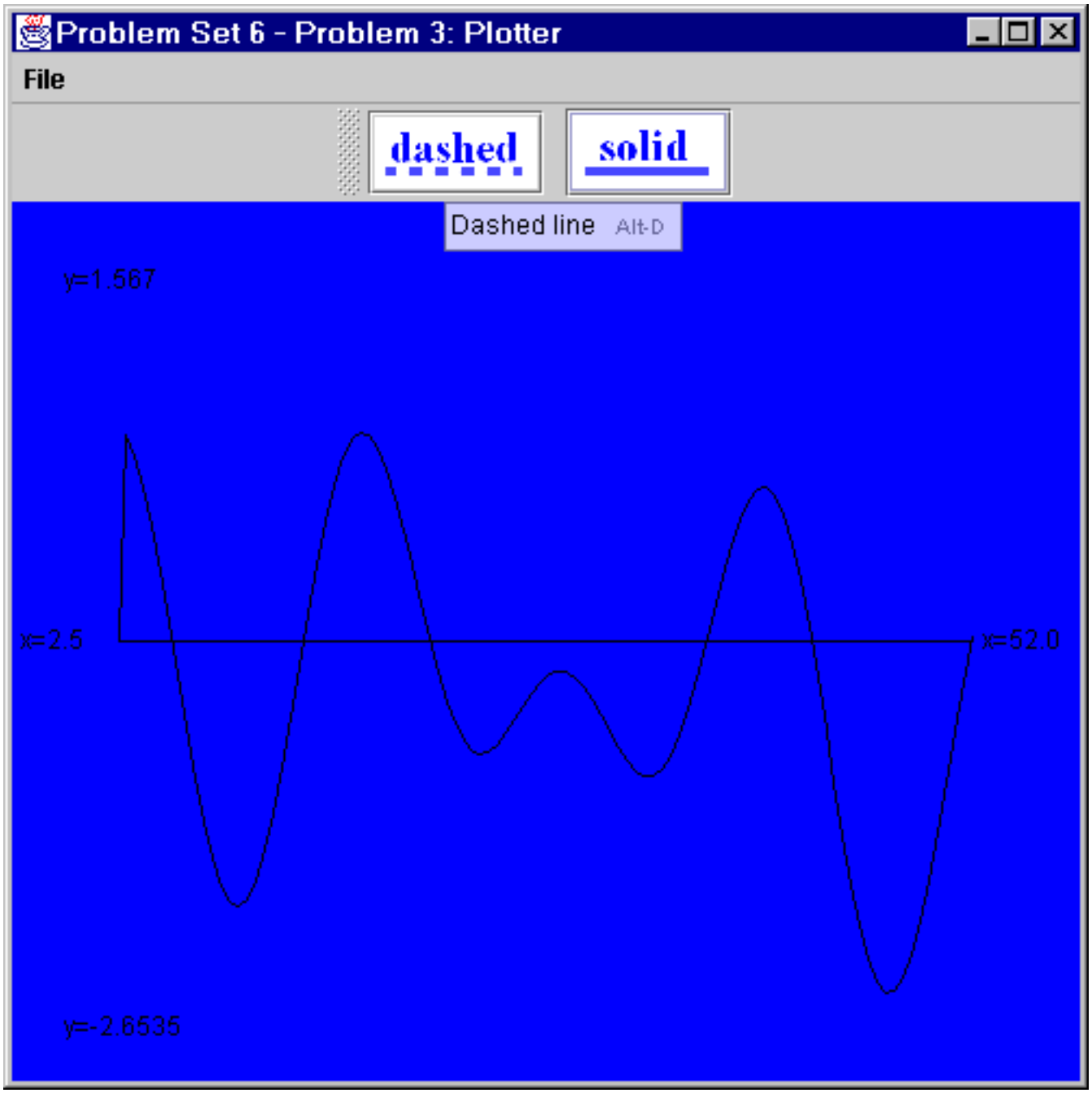
- "Import Data": which should open the file *data6_3* and read in the x and y coordinates. The first number in that file is the number of points followed by that number of couples of values for x and y coordinates.
- "Exit": which should terminate the execution of the program.

Your application should have also a toolbar with the following two buttons:

- "dashes": plotting using a dash line
- "solid": plotting using a solid line

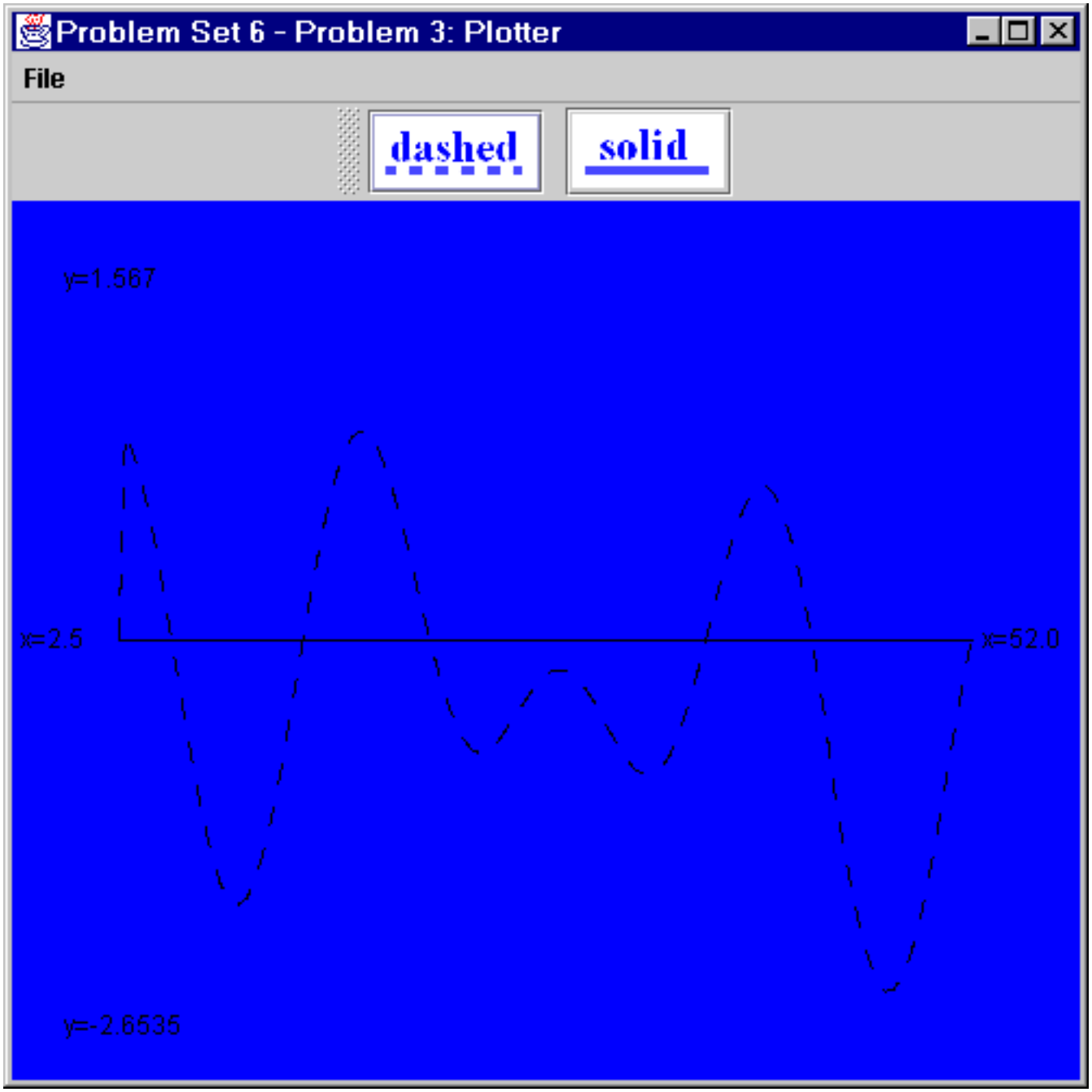
Finally, tooltips should appear above the buttons to inform the user of their purpose. For example

when the mouse is left above the button for the dash line the program provides the tooltip as shown below:



After importing in the data the program should print the x and y points as a polyline (Java 2D) as shown above. It should print the minimum and maximum values of x and y as shown.

Selecting either using the button ("dashed") the program should use a dashed line and look as follows:



You should submit all files of your Java source-code files that implement this program as well as the *data6_1* file. The *main()* should be provided in a file named ***ps6_1.java*** and be compiled using the command: *javac ps6_1.java*

You should also submit screendumps from the execution of your program showing the graph plotted.

The image files for the buttons are provided using cvs, but your are free to use any other images that make sense on the buttons.

Problem 2: [40%]

In this problem you need to write a multithreaded application to handle the balances of credit cards. You need to provide the source code in the following files:

CreditCardAccount:

In this file you should provide the definition of the class CreditCardAccount. This class should have among others the following private data members:

- *creditID*: an int to store the credit card number
- *currentBalance*: a double to store the current balance
- *allowableLimit*: a double to store the allowable limit

You may need to have other data members, which also should be specified as private.

The class should have among other methods that you may need to write the following 2 methods:

- *charge(double amount)*: which should charge the credit card by adding to the current balance the provided amount if and only if the sum after the addition does not exceed the allowable limit. It should write to a file with name the character 'c' followed by the *creditID* the following information:
  - The amount that has been either charged or denied authorization
  - The information of the account: Account: *creditID*, *currentBalance*, and *allowableLimit*
  - The thread that has processed the transaction and its threadgroup

- *payment(double amount)*: which should credit the provided amount by subtracting it from the current balance. It should write to a file with name the character 'c' followed by the *creditID* the following information:
  - The amount that has been credited
  - The information of the account: Account: *creditID*, *currentBalance*, and *allowableLimit*
  - The thread that has processed the transaction and its threadgroup

CreditCardTranscationsThread:

In this file you should provide the class *CreditCardTranscationsThread*, which should extend the Thread class. It should have only one private member data the account , a *CreditCardAccount* object, with which it is associated.

Its *run* method should have a for loop with 10 iterations which should sleep for a random number of milliseconds given by (100*Math.random()). Then, it should charge the account, by calling the corresponding method, an amount equal to a random number (e.g. Math.random()) times the allowableLimit of the account. Finally, it should credit the card, by invoking the payment method, an amount equal to a random number (e.g. Math.random()) times the currentBalance of the account.

CreditCardTransactionsRunnable:

In this file you should provide the class *CreditCardTransactionsRunnable*, which should implement the *Runnable* interface. This class should have the same functionality as the class *CreditCardTranscationsThread* above. It should have only one private member data the account , a *CreditCardAccount* object, with which it is associated.

Its *run* method should have a for loop with 10 iterations which should sleep for a random number of milliseconds given by (100*Math.random()). Then, it should charge the account, by calling the corresponding method, an amount equal to a random number (e.g. Math.random()) times the allowableLimit of the account. Finally, it should credit the card, by invoking the payment method, an amount equal to a random number (e.g. Math.random()) times the currentBalance of the account.

PS6_2:

You should provide the *main()* method in this file. First two credit card accounts, named x1 and x2, should be created with credit card numbers 32483273 and 93455454, current balances 0.00 and 250.75 and allowable limits 10,000.00 and 5,000.00, respectively. Having created the new accounts, the method should print them out.

Then, a ThreadGroup named "Thread Group X" should be created.

Then, two CreditCardTranscationsThread objects, named t1 and t2, should be defined. Their threadgroup should be set to be the previously defined threadgroup. Two CreditCardTransactionsRunnable, named r1 and r2, should also be defined and used to define two threads, named t3 and t4. Threads t1 and t3 should be associated with the account x1 and threads t2 and t4 with the account x2.

After printing out the defined threads and their threadgroups you should start the threads.

General note: All amounts should be written with the thousands separated by a comma and with only

two decimal digits printed representing the cents.

Sample output from the execution of the completed program:

>java Sol6_2

Newly created credit card account:
Credit card number = 32483273 Current Balance = 0.00 Limit = 10,000.00

Newly created credit card account:
Credit card number = 93455454 Current Balance = 250.75 Limit = 5,000.00

ThreadGroup: java.lang.ThreadGroup[name=Thread Group X,maxpri=10]
Thread: Thread[Thread t1,5,Thread Group X]
Thread: Thread[Thread t2,5,Thread Group X]

ThreadGroup: java.lang.ThreadGroup[name=main,maxpri=10]
Thread: Thread[Thread-0,5,main]
Thread: Thread[Thread-1,5,main]

Also the file ***c93455454*** looks like:

Account has been charged an amount of $3,831.47
Account: Credit card number = 93455454 Current Balance = 4,082.22 Limit = 5,000.00
By the Thread: Thread[Thread-1,5,main]
of the ThreadGroup: java.lang.ThreadGroup[name=main,maxpri=10]
The amount $4,590.53 was not authorized to be charged
Account: Credit card number = 93455454 Current Balance = 4,082.22 Limit = 5,000.00
By the Thread: Thread[Thread t2,5,Thread Group X]
of the ThreadGroup: java.lang.ThreadGroup[name=Thread Group X,maxpri=10]
A payment of 2,597.74 has been made
Account: Credit card number = 93455454 Current Balance = 1,484.48 Limit = 5,000.00
By the Thread: Thread[Thread-1,5,main]
of the ThreadGroup: java.lang.ThreadGroup[name=main,maxpri=10]
A payment of 1,349.31 has been made
Account: Credit card number = 93455454 Current Balance = 135.17 Limit = 5,000.00
By the Thread: Thread[Thread t2,5,Thread Group X]
of the ThreadGroup: java.lang.ThreadGroup[name=Thread Group X,maxpri=10]
Account has been charged an amount of $1,485.69
Account: Credit card number = 93455454 Current Balance = 1,620.86 Limit = 5,000.00
By the Thread: Thread[Thread-1,5,main]

of the ThreadGroup: java.lang.ThreadGroup[name=main,maxpri=10]

..... etc.

and the file c32483273:

Account has been charged an amount of $6,508.41
Account: Credit card number = 32483273 Current Balance = 6,508.41 Limit = 10,000.00
By the Thread: Thread[Thread-0,5,main]
of the ThreadGroup: java.lang.ThreadGroup[name=main,maxpri=10]
Account has been charged an amount of $2,381.95
Account: Credit card number = 32483273 Current Balance = 8,890.36 Limit = 10,000.00
By the Thread: Thread[Thread t1,5,Thread Group X]
of the ThreadGroup: java.lang.ThreadGroup[name=Thread Group X,maxpri=10]
A payment of 1,323.96 has been made
Account: Credit card number = 32483273 Current Balance = 7,566.39 Limit = 10,000.00
By the Thread: Thread[Thread-0,5,main]
of the ThreadGroup: java.lang.ThreadGroup[name=main,maxpri=10]
A payment of 6,147.78 has been made
Account: Credit card number = 32483273 Current Balance = 1,418.61 Limit = 10,000.00
By the Thread: Thread[Thread t1,5,Thread Group X]
of the ThreadGroup: java.lang.ThreadGroup[name=Thread Group X,maxpri=10]

You need to provide the above described classes in the corresponding files:

- ***CreditCardAccount.java***
- ***CreditCardTransactionsRunnable.java***
- ***CreditCardTranscationsThread.java***
- ***PS6_2.java***

Note:

Please submit **both** printouts of the source code you have written (preferably using *% enscript -2Gr -Pprinter filename*) and (or screen dumps of) the execution output (using *%xdpr -Pprinter*), with your name and username clearly written on the first page of the **stapled** submitted problem set. The submitted code must be identical to that electronically turned in (as described above).
