# Summer 2017.

# **Aid Management Application (AMA)**

**Milestone 6: the Aid Management ApPlication**

For your final milestone for this project, create a class called "AidApp".   
  
AidApp is a class that uses the previously created classes in this project to give the user the capability to store and retrieve Perishable and Non-Perishable product information from and to a file.   
  
The AidApp class has several private member functions and only two public functions.  
  
A description for each function is provided below. Those that are more complex will be complimented with suggested pseudo code to help you implement the function. You may use the pseudo code as suggested, revise it to improve on the logic, or implement your own logic.  
  
Code the AidApp class in files **AidApp.h** and **AidApp.cpp**.

## **AidApp class**

### Private member variables:

**char filename\_[256];**

Holds the name of the text file used to store the product information.

**Product\* product\_[MAX\_NO\_RECS];**

An array of Product pointers, MAX\_NO\_RECS long. (i.e. Each element of this array is a product pointer).

**fstream datafile\_;**

An fstream instance used to create and access a file.

**int noOfProducts\_;**

Number of Products (perishable or non-perishable) pointed to by the product\_ array.

### The Constructor

The AidApp constructor receives a constant char string called filename and then:   
1- Copies filename to filename\_ member variable  
2- Sets all the product\_ elements to nullptr  
3- Sets noOfProducts\_ to zero  
4- Loads the Records (calls the member function to do this)

### Private member functions:

#### Copy and assignment prevention

Make sure the AidApp cannot get copied or assigned to another AidApp.

**void pause()const;**

Prints: "Press Enter to continue..."<NEWLINE> then waits for the user to hit enter. If the user hits any other key, the key is ignored. Only the ENTER key will terminate this function.

**int menu();**

Menu() displays the menu as follows and waits for the user to select an option.

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> \_

**^** The cursor should be positioned here when the menu is printed

- If the selection is valid, menu() will return the selection otherwise it will return -1  
- The standard input buffer (keyboard) must be cleared before the function exits.

**void loadRecs();**

Opens the data file for reading. If the file does not exist, it will create an empty file, close the file and exit.

Otherwise, read the data file and store each record to the pointer array of Products accordingly (overwriting the old ones).

Note: Prevent memory leaks by deleting each Product element pointer in the array before overwriting/replacing the element with a new Product pointer.

After reading all the records, close the file.

Pseudo code:

Set readIndex to zero

Open the file for reading (use ios::in)

if the file is in fail state it means there is no file on the disk, then

clear the failure

close the file

open the file for writing (ios::out) to create the file

close thefile

otherwise

until reading fails loop

deallocate the memory pointed by product pointer at readindex

read one char character to identify type of Product into Id character

if Id character is P

Dynamically create a Perishable product and hold it in product pointer at readIndex

if Id character is N

Dynamically create a Non-perishable product and hold it in product pointer at readIndex

if either P or N is read

skip the comma in the file

load the product from the file (using its load method)

add one to read index

continue the loop

set number of products to readIndex

close the datafile

**void saveRecs();**

- Opens the data file for writing  
- Loops through the product\_ array up to noOfProducts\_ and stores them in the datafile\_  
- Closes the file

**void listProducts()const;**

- Prints the following title:

Row | SKU | Product Name | Cost | QTY| Unit |Need| Expiry

-----|--------|--------------------|-------|----|----------|----|----------

- Then loops through the products\_ array up to noOfProducts\_ and prints the row number (four spaces wide, right justified), followed by a bar/pipe character (|) surrounded by two spaces  
- Then prints the current Product in the loop followed by a newline  
- When the number of printed items reaches 10, it will pause until the user hits the Enter key.  
- With each loop iteration, it will calculate the total cost of the products in a double value using the operator+= implemented by the Product class.  
- When the list is done, a line of dashes will be printed:

---------------------------------------------------------------------------

- Finally, the total cost will be printed:

Total cost of support: $9999.99

The total cost value is printed with a dollar sign at left and two digits after decimal point

**int SearchProducts(const char\* sku)const;**

Loops through the product\_ up to noOfProducts\_ and checks each of them for the same SKU as the incoming argument using the operator== implemented by the Product class.  
If a match is found it will return the index of the found Product in the product\_ array, otherwise it will return -1.

**void addQty(const char\* sku);**

Updates the quantity on hand for a Product.  
updateQty() searches for the Product with the same sku as the incoming argument. If not found it will display:

"Not found!"<NEWLINE>

If found, it will display the Product in non-linear format and then asks for an integer for quantity purchased:

"Please enter the number of purchased items: "

If it cannot read the integer it prints:

"Invalid quantity value! "<NEWLINE>

If it can read the integer, it makes sure the amount is less than or equal to the amount required (i.e. less than qtyNeeded() - quantity()). If it is less than or equal, it will add the value to the quantity on hand of the product using operator+= implemented by the Product class. If the value is not less than or equal the amount needed, it will only accept the amount required and prints a message to return the extra:

"Too many items; only 999 is needed, please return the extra 99 items. "< NEWLINE >

Lastly, all records will be saved back to the file and a message displayed:

"Updated!" <NEWLINE>

Make sure after the entry the keyboard is flushed.

**void addProduct(bool isPerishable);**

Depending on the value of the argument being true or false, create a Perishable or Non-perishable Product and get the values from the user and add it to the end of the product\_ array and save the records (call saveRecs()). If there is an error, display the Product and exit the function (this will show the error message).

**int run();**

Display the menu, receive the user’s selection, and do the action requested (follow with a pause using the pause() function), and repeat (redisplay the menu…) until the user selects zero to exit.   
1- List products

List the products.

2- Display product

Ask for a sku using this prompt  
"Please enter the SKU: "  
(receive input from console) then search for the Product.

If found, display the Product information in non-liner format

Otherwise display:

"Not found!"

3- Add non-perishable product

Add a Non-Perishable product to the system using the   
 addProduct function

Load the records.

4- Add perishable product

Add a Perishable product to the to the system using the

addProduct function

Load the records.

5- Add to quantity of purchased products

Ask for a sku using this prompt  
"Please enter the SKU: "  
(receive input from console) and then add to the quantity using the addQty() function.

0- Exit program

The program will terminate printing:  
 "Goodbye!!"

In case of invalid menu selection the program will print:

"===Invalid Selection, try again.==="

Followed by a pause (pause() function) before redisplaying the menu.

The Run function will return 0 when it ends.

**Project SUBMISSION**

**comments.txt:**

Create a file called comments.txt to be submitted along with your project.   
Add the following information to comments.txt:

Citation:

If you have used any code or logic developed by others in your solution, you must identify the source of that code or logic in comments in your source code.

You must insert these comments at the place in your source code where the code/logic is being used.

You must also copy and list this information in comments.txt.

*Failure to identify any code or logic developed by others and included in your solution will result in an assessment of plagiarism by your professor.*

Enhancements:

If you have enhanced your solution with any features not in the specifications, you should identify them clearly through comments in your source code.

You should also copy and list this information in comnments.txt.

**SUBMISSION methods**

You have 3 choices for submitting your project:

1. **Short**  
   You can select this option if you have already successfully submitted milestones 1 to 5
2. **Long**

You can select this option if you have not submitted the past 5 milestones and want to only submit the Final Assembly. Your program must work exactly as described in the past 5 milestones.

1. **Open**  
   This submission does not test the output of your application but captures and submits it to your professor.

You can select this option if you believe your project works properly but you could not match the exact output requested in either Short or Long submissions.

YOU are responsible to do all the tests asked in either Short or Long submission. If you miss any steps of those tests your submission may be rejected and you will have to re-submit your application incurring more deductions for having to resubmit and not being on time.

Also, you must **add a note in the comments.txt** file with an explanation why you have chosen the open submission rather than one of the other two.

**SHort sUBMISSION:**

If not on matrix already, upload your project files and **comments.txt** to your matrix account. Compile your code as follows:  
**> g++ -Wall -std=c++0x -o fp AidApp.cpp AmaPerishable.cpp AmaProduct.cpp Date.cpp ErrorMessage.cpp Product.cpp aidAppMai<ENTER>**

This command will compile your code and name your executable “**fp**”

Execute fp and make sure everything works properly.

Finally run the following script from your account: (replace profname.proflastname with your professors Seneca userid)

**~profname.proflastname/submit 244\_fp\_short <ENTER>**

and run your application using the RED underlined **bold** *italic* values in the short output section below.

Please note that a successful submission does not guarantee full credit for this workshop.

If the professor is not satisfied with your implementation, your professor may ask you to resubmit. Resubmissions will attract a penalty.

**Output (short submission)**

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **1**

Row | SKU | Product Name | Cost | QTY| Unit |Need| Expiry

-----|--------|--------------------|-------|----|----------|----|----------

1 | 132 |Banana | 0.99| 6|Kilos | 10|2015/05/13

2 | 122 |Bucket | 11.30| 150| | 300|

3 | 124 |Corn | 4.00| 0|Packs | 140|2018/04/03

4 | 200 |Sand Bags | 3.38| 20|packs |1000|

5 | 5678 |Sugar | 6.78| 0|1kg Bags | 200|

6 | 111 |Water Container | 12.79| 500|packs |5000|

7 | 1111 |Milk | 3.99| 10|bag | 100|2017/09/20

---------------------------------------------------------------------------

Total cost of support: $8204.21

Press Enter to continue...

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **2**

Please enter the SKU: **132**

Sku: 132

Name: Banana

Price: 0.99

Price after tax: N/A

Quantity On Hand: 6 Kilos

Quantity Needed: 10

Expiry date: 2015/05/13

Press Enter to continue...

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **2**

Please enter the SKU: **1111**

Sku: 1111

Name: Milk

Price: 3.99

Price after tax: N/A

Quantity On Hand: 10 bag

Quantity Needed: 100

Expiry date: 2017/09/20

Press Enter to continue...

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **3**

Sku: **1234**

Name: **Tent**

Unit:

Taxed? (y/n): **y**

Price: **120.33**

Quantity On hand: **20**

Quantity Needed: **200**

Product added

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **4**

Sku: **2345**

Name: **Flu Vaccine**

Unit: **Boxes**

Taxed? (y/n): **n**

Price: **12.34**

Quantity On hand: **0**

Quantity Needed: **5000**

Expiry date (YYYY/MM/DD): **2018/12/12**

Product added

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **1**

Row | SKU | Product Name | Cost | QTY| Unit |Need| Expiry

-----|--------|--------------------|-------|----|----------|----|----------

1 | 132 |Banana | 0.99| 6|Kilos | 10|2015/05/13

2 | 122 |Bucket | 11.30| 150| | 300|

3 | 124 |Corn | 4.00| 0|Packs | 140|2018/04/03

4 | 200 |Sand Bags | 3.38| 20|packs |1000|

5 | 5678 |Sugar | 6.78| 0|1kg Bags | 200|

6 | 111 |Water Container | 12.79| 500|packs |5000|

7 | 1111 |Milk | 3.99| 10|bag | 100|2017/09/20

8 | 1234 |Tent | 135.97| 20| | 200|

9 | 2345 |Flu Vaccine | 12.34| 0|Boxes |5000|2018/12/12

---------------------------------------------------------------------------

Total cost of support: $10923.67

Press Enter to continue...

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **5**

Please enter the SKU: **1234**

Sku: 1234

Name: Tent

Price: 120.33

Price after tax: 135.97

Quantity On Hand: 20

Quantity Needed: 200

Please enter the number of purchased items: **100**

Updated!

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **1**

Row | SKU | Product Name | Cost | QTY| Unit |Need| Expiry

-----|--------|--------------------|-------|----|----------|----|----------

1 | 132 |Banana | 0.99| 6|Kilos | 10|2015/05/13

2 | 122 |Bucket | 11.30| 150| | 300|

3 | 124 |Corn | 4.00| 0|Packs | 140|2018/04/03

4 | 200 |Sand Bags | 3.38| 20|packs |1000|

5 | 5678 |Sugar | 6.78| 0|1kg Bags | 200|

6 | 111 |Water Container | 12.79| 500|packs |5000|

7 | 1111 |Milk | 3.99| 10|bag | 100|2017/09/20

8 | 1234 |Tent | 135.97| 120| | 200|

9 | 2345 |Flu Vaccine | 12.34| 0|Boxes |5000|2018/12/12

---------------------------------------------------------------------------

Total cost of support: $24520.96

Press Enter to continue...

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **0**

Goodbye!!

**LOng sUBMISSION:**

If not on matrix already, upload your project files and **comments.txt** to your matrix account. Compile your code as follows:  
**> g++ -Wall -std=c++0x -o fp Date.cpp Error.cpp Item.cpp  
 NonPerishable.cpp Perishable.cpp PosApp.cpp PosAppMain.cpp<ENTER>**

This command will compile your code and name your executable “**fp**”

Execute fp and make sure everything works properly.

Finally run the following script from your account: (replace profname.proflastname with your professors Seneca userid)

**~profname.proflastname/submit 244\_fp\_long <ENTER>**

and run your application using the RED underlined **bold** *italic* values in the short output section below.

Please note that a successful submission does not guarantee full credit for this workshop.

If the professor is not satisfied with your implementation, your professor may ask you to resubmit. Resubmissions will attract a penalty.

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **1**

Row | SKU | Product Name | Cost | QTY| Unit |Need| Expiry

-----|--------|--------------------|-------|----|----------|----|----------

1 | 132 |Banana | 0.99| 6|Kilos | 10|2015/05/13

2 | 122 |Bucket | 11.30| 150| | 300|

3 | 124 |Corn | 4.00| 0|Packs | 140|2018/04/03

4 | 200 |Sand Bags | 3.38| 20|packs |1000|

5 | 5678 |Sugar | 6.78| 0|1kg Bags | 200|

6 | 111 |Water Container | 12.79| 500|packs |5000|

7 | 1111 |Milk | 3.99| 10|bag | 100|2017/09/20

---------------------------------------------------------------------------

Total cost of support: $8204.21

Press Enter to continue...

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **-1**

===Invalid Selection, try again.===

Press Enter to continue...

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **6**

===Invalid Selection, try again.===

Press Enter to continue...

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **2**

Please enter the SKU: **abc**

Not found!

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **2**

Please enter the SKU: **132**

Sku: 132

Name: Banana

Price: 0.99

Price after tax: N/A

Quantity On Hand: 6 Kilos

Quantity Needed: 10

Expiry date: 2015/05/13

Press Enter to continue...

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **2**

Please enter the SKU: **1111**

Sku: 1111

Name: Milk

Price: 3.99

Price after tax: N/A

Quantity On Hand: 10 bag

Quantity Needed: 100

Expiry date: 2017/09/20

Press Enter to continue...

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **3**

Sku: **1234**

Name: **Tent**

Unit:

Taxed? (y/n): **y**

Price: **120.33**

Quantity On hand: **20**

Quantity Needed: **200**

Product added

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **1**

Row | SKU | Product Name | Cost | QTY| Unit |Need| Expiry

-----|--------|--------------------|-------|----|----------|----|----------

1 | 132 |Banana | 0.99| 6|Kilos | 10|2015/05/13

2 | 122 |Bucket | 11.30| 150| | 300|

3 | 124 |Corn | 4.00| 0|Packs | 140|2018/04/03

4 | 200 |Sand Bags | 3.38| 20|packs |1000|

5 | 5678 |Sugar | 6.78| 0|1kg Bags | 200|

6 | 111 |Water Container | 12.79| 500|packs |5000|

7 | 1111 |Milk | 3.99| 10|bag | 100|2017/09/20

8 | 1234 |Tent | 135.97| 20| | 200|

---------------------------------------------------------------------------

Total cost of support: $10923.67

Press Enter to continue...

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **4**

Sku: **a**

Name: **a**

Unit: **a**

Taxed? (y/n): **a**

Only (Y)es or (N)o are acceptable

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **4**

Sku: **a**

Name: **a**

Unit: **a**

Taxed? (y/n): **y**

Price: **a**

Invalid Price Entry

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **4**

Sku: **a**

Name: **a**

Unit: **a**

Taxed? (y/n): **y**

Price: **1**

Quantity On hand: **a**

Invalid Quantity Entry

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **4**

Sku: **a**

Name: **a**

Unit: **a**

Taxed? (y/n): **y**

Price: **1**

Quantity On hand: **1**

Quantity Needed: **a**

Invalid Quantity Needed Entry

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **4**

Sku: **a**

Name: **a**

Unit: **a**

Taxed? (y/n): **y**

Price: **1**

Quantity On hand: **1**

Quantity Needed: **1**

Expiry date (YYYY/MM/DD): **12/12/12**

Invalid Year in Date Entry

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **4**

Sku: **a**

Name: **a**

Unit: **a**

Taxed? (y/n): **y**

Price: **1**

Quantity On hand: **1**

Quantity Needed: **1**

Expiry date (YYYY/MM/DD): **2010/13/10**

Invalid Month in Date Entry

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **4**

Sku: **a**

Name: **a**

Unit: **a**

Taxed? (y/n): **y**

Price: **1**

Quantity On hand: **1**

Quantity Needed: **1**

Expiry date (YYYY/MM/DD): **2010/10/100**

Invalid Day in Date Entry

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **4**

Sku: **2345**

Name: **Flu Vaccine**

Unit: **Boxes**

Taxed? (y/n): **n**

Price: **12.34**

Quantity On hand: **0**

Quantity Needed: **5000**

Expiry date (YYYY/MM/DD): **2018/12/12**

Product added

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **1**

Row | SKU | Product Name | Cost | QTY| Unit |Need| Expiry

-----|--------|--------------------|-------|----|----------|----|----------

1 | 132 |Banana | 0.99| 6|Kilos | 10|2015/05/13

2 | 122 |Bucket | 11.30| 150| | 300|

3 | 124 |Corn | 4.00| 0|Packs | 140|2018/04/03

4 | 200 |Sand Bags | 3.38| 20|packs |1000|

5 | 5678 |Sugar | 6.78| 0|1kg Bags | 200|

6 | 111 |Water Container | 12.79| 500|packs |5000|

7 | 1111 |Milk | 3.99| 10|bag | 100|2017/09/20

8 | 1234 |Tent | 135.97| 20| | 200|

9 | 2345 |Flu Vaccine | 12.34| 0|Boxes |5000|2018/12/12

---------------------------------------------------------------------------

Total cost of support: $10923.67

Press Enter to continue...

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **5**

Please enter the SKU: **1234**

Sku: 1234

Name: Tent

Price: 120.33

Price after tax: 135.97

Quantity On Hand: 20

Quantity Needed: 200

Please enter the number of purchased items: **100**

Updated!

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **1**

Row | SKU | Product Name | Cost | QTY| Unit |Need| Expiry

-----|--------|--------------------|-------|----|----------|----|----------

1 | 132 |Banana | 0.99| 6|Kilos | 10|2015/05/13

2 | 122 |Bucket | 11.30| 150| | 300|

3 | 124 |Corn | 4.00| 0|Packs | 140|2018/04/03

4 | 200 |Sand Bags | 3.38| 20|packs |1000|

5 | 5678 |Sugar | 6.78| 0|1kg Bags | 200|

6 | 111 |Water Container | 12.79| 500|packs |5000|

7 | 1111 |Milk | 3.99| 10|bag | 100|2017/09/20

8 | 1234 |Tent | 135.97| 120| | 200|

9 | 2345 |Flu Vaccine | 12.34| 0|Boxes |5000|2018/12/12

---------------------------------------------------------------------------

Total cost of support: $24520.96

Press Enter to continue...

Disaster Aid Supply Management Program

1- List products

2- Display product

3- Add non-perishable product

4- Add perishable product

5- Add to quantity of purchased products

0- Exit program

> **0**

Goodbye!!

**Open sUBMISSION:**

In “**comments.txt**”, explain why you have chosen the open submission rather than one of the other two.

If not on matrix already, upload your project files and **comments.txt** to your matrix account. Compile your code as follows:  
**> g++ -Wall -std=c++0x -o fp Date.cpp Error.cpp Item.cpp  
 NonPerishable.cpp Perishable.cpp PosApp.cpp PosAppMain.cpp<ENTER>**

This command will compile your code and name your executable “**fp**”

Execute fp and make sure everything works properly.

Finaly run the following script from your account: (replace profname.proflastname with your professors Seneca userid)

**~profname.proflastname/submit 244\_fp\_open <ENTER>**

and run your application using the RED underlined **bold** *italic* values in the ***long output section*** above.

Please note that a successful submission does not guarantee full credit for this workshop.

If the professor is not satisfied with your implementation, your professor may ask you to resubmit. Resubmissions will attract a penalty.