Lab 1

Jiho Kim, Garth Slaney

ENSF 480

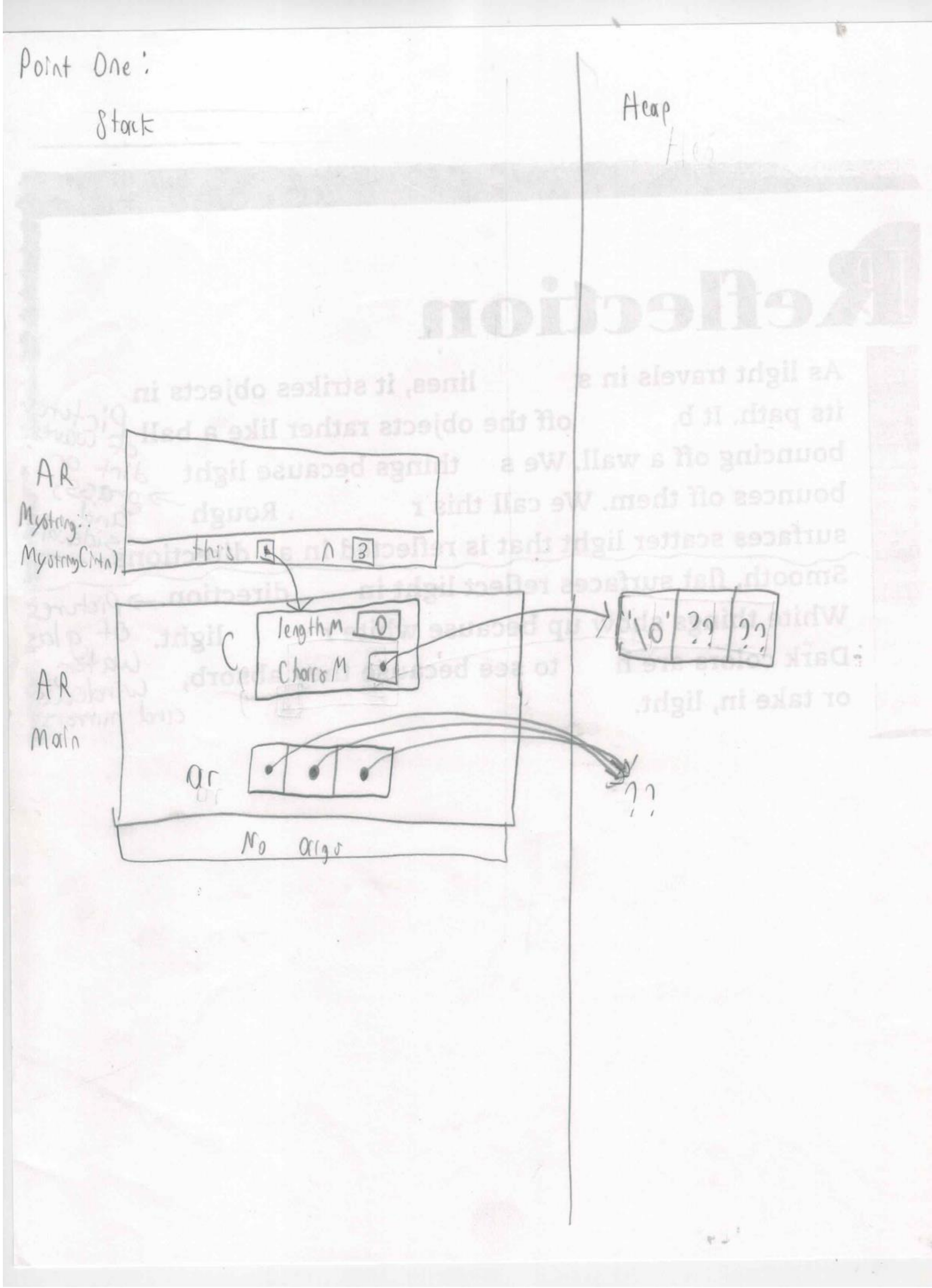
Submission Date: September 24, 2020

University of Calgary

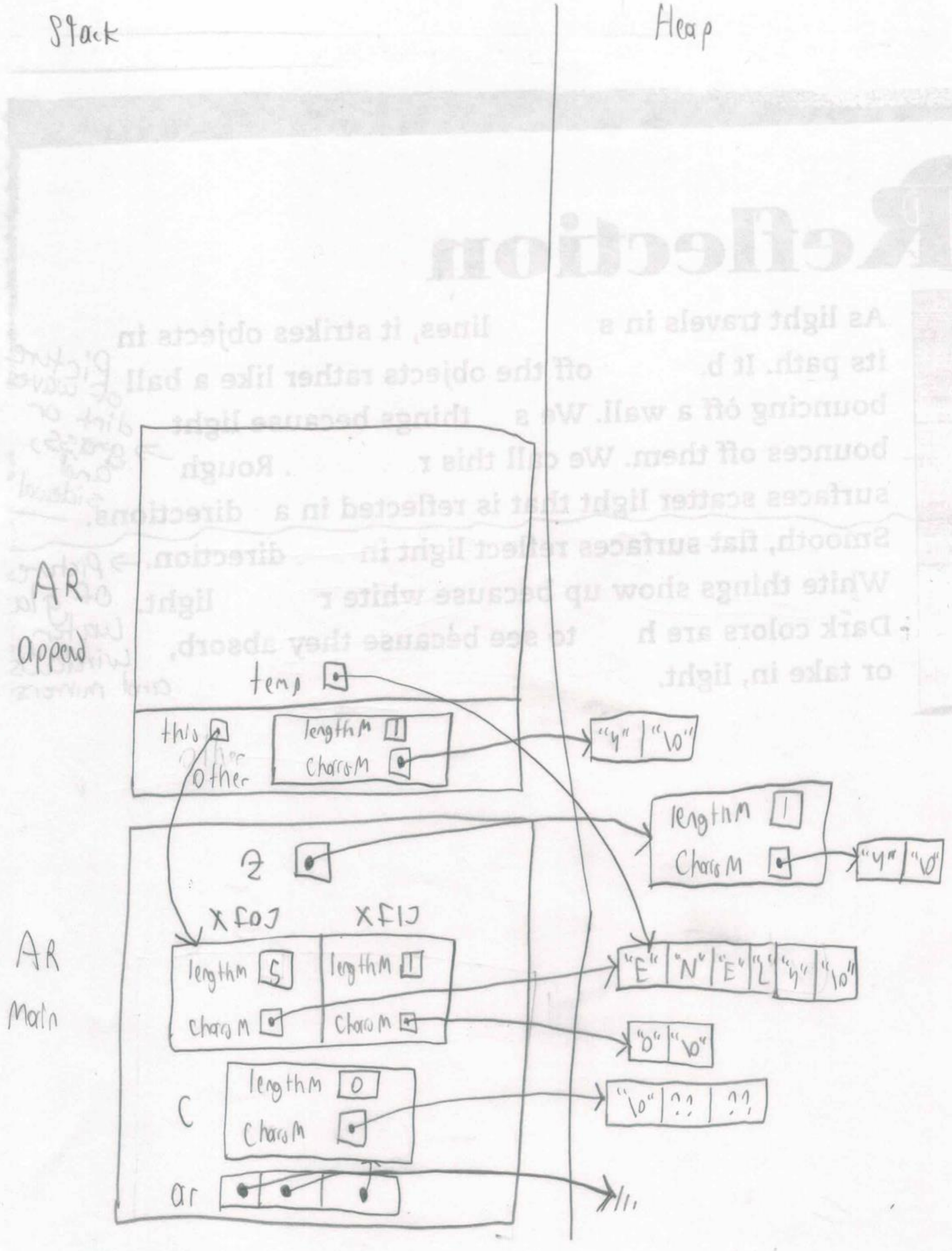
Answer the questions in Exercise A in the following table and post it into the D2L

Program output and its order	Your explanation (why and where is the cause for this output)
constructor with int argument is called.	it is called at line 12 in exAmain. The statement, Mystring c = 3is interpretedby the compiler as a call to theconstructor Mystring::Mystring(int n).
default constructor is called. default constructor is called.	It is called in line 18 in exAmain. The statement, Mystring x[2] is interpreted by the compiler as two calls to the default constructor Mystring::Mystring().
constructor with char* argument is called.	It is called in line 22 in exAmain. The statement, Mystring* z = new Mystring("4") is interpreted by the compiler as a call to Mystring::Mystring(const char *s).
copy constructor is called. copy constructor is called.	It is called in line 24 in exAmain. The statement, x[0].append(*z).append(x[1]) is interpreted by the complier as two calls to Mystring::Mystring(const Mystring& source) to copy the arguments need for Mystring& Mystring::append(const Mystring other).
destructor is called. destructor is called.	It is called in line 24 in exAmain. The statement, x[0].append(*z).append(x[1]) is interpreted by the complier as two calls to Mystring::~Mystring() to destroy the arguments for Mystring& Mystring::append(const Mystring other).
copy constructor is called.	It is called in line 26 in exAmain. The statement Mystring mars = x[0] is interpreted by the compiler as a call to Mystring::Mystring(const Mystring& source).
assignment operator called.	It is called in line 28 in exAmain. The statement x[1] = x[0] is interpreted by the compiler as a call to Mystring& Mystring::operator =(const Mystring& S).
constructor with char* argument is called. constructor with char* argument is called.	It is called in line 30 and line 32 in exAmain. The statements Mystring jupiter("White") and ar[0] = new Mystring ("Yellow") are interpreted as calls to Mystring::Mystring(const char *s).
destructor is called. destructor is called. destructor is called. destructor is called. destructor is called.	They are called in line 34 in exAmain. When leaving the block of code the destructors are called for all Mystring objects declared in that block.
constructor with char* argument is called.	It is called in line 39 in exAmain. The statement Mystring d = "Green" Is interpreted as a call to Mystring::Mystring(const char *s).
Program terminated successfully.	It is called in line 41 from the print statement.
destructor is called. destructor is called	It is called in line 43 as the Mystring objects declared in the scope of the code are destroyed.

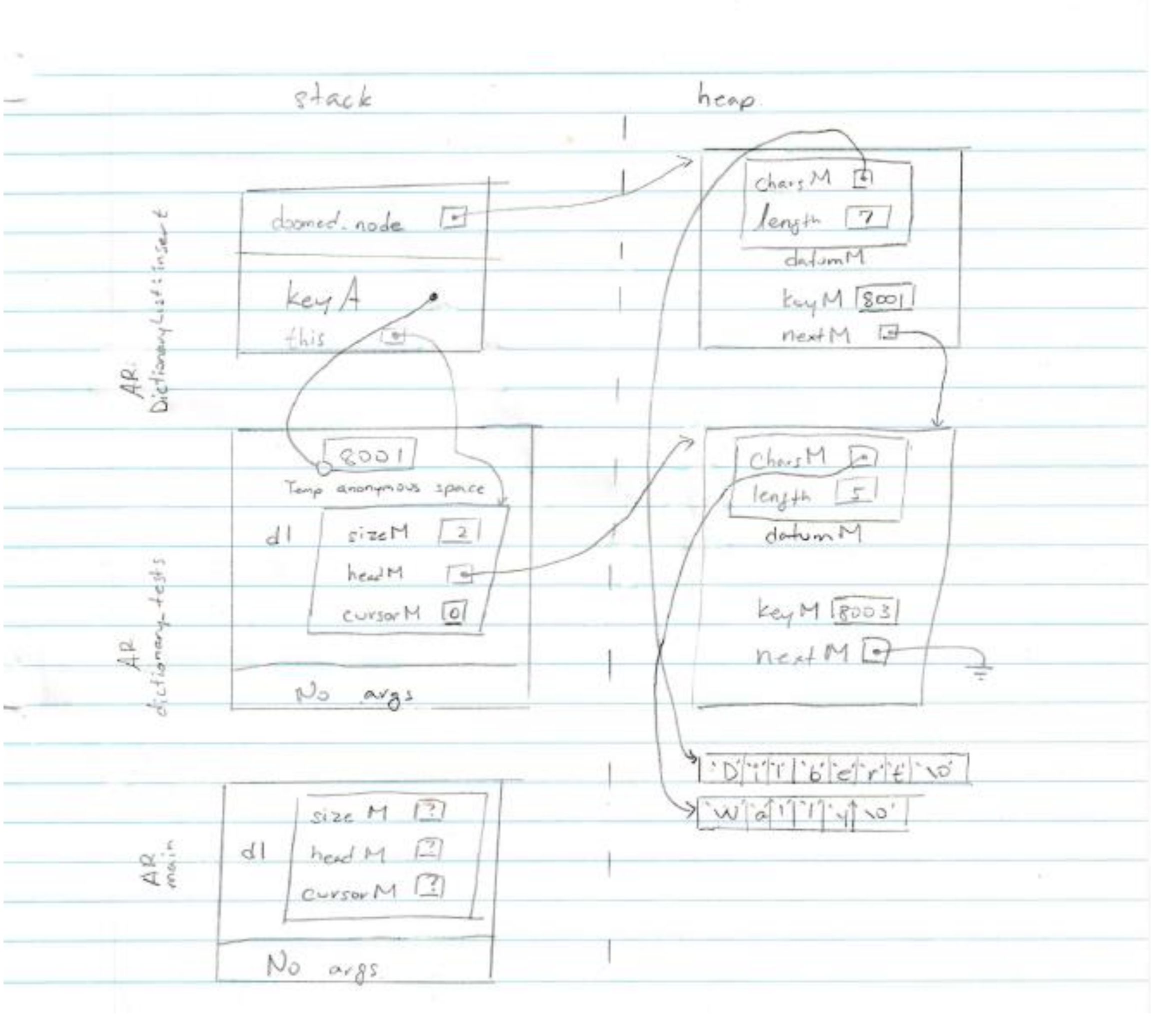
Exercise A AR Diagrams



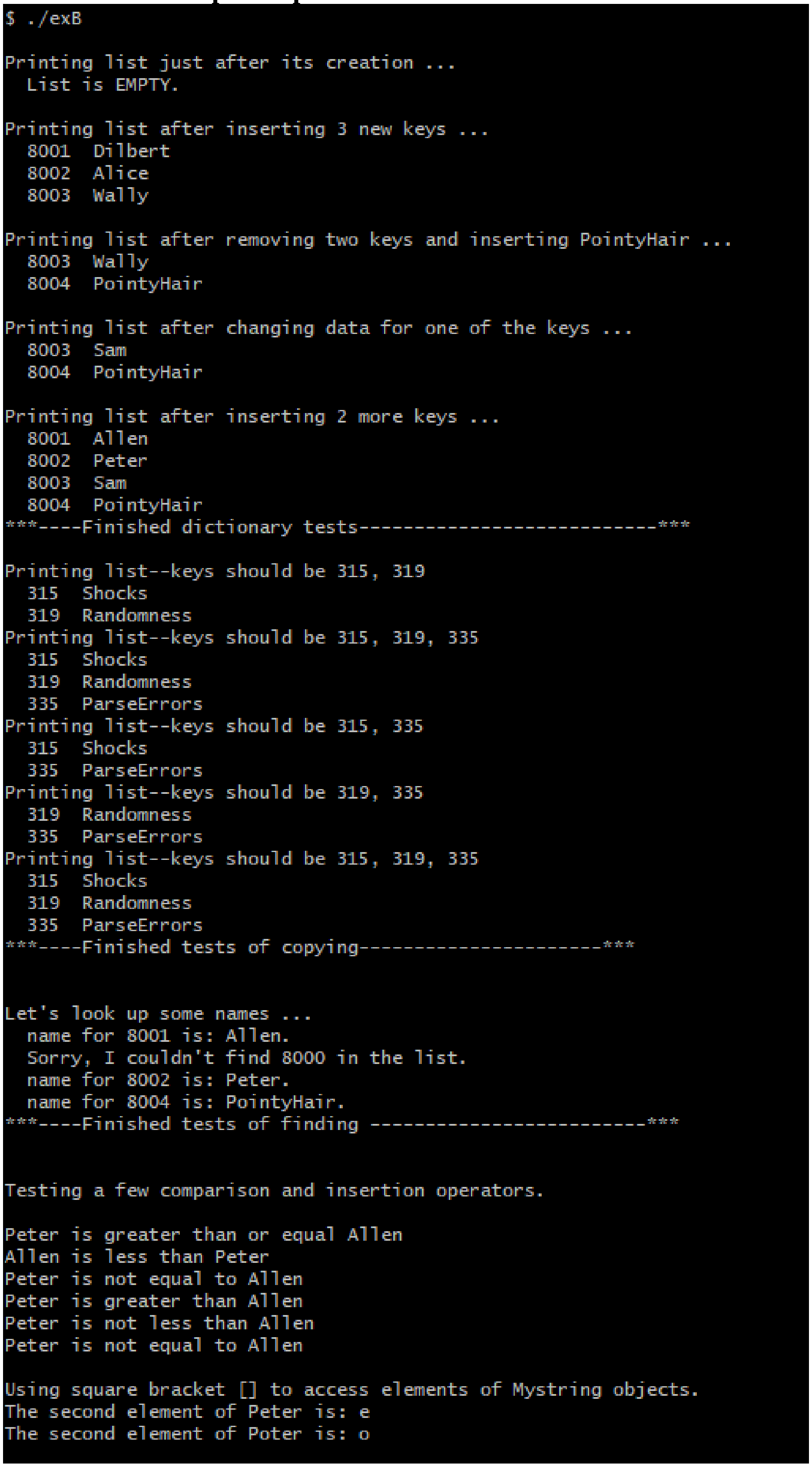
Point two:


Stack
Heap
AR
append
temp
this
other
lengthM 1
CharsM
"y" "\0"
lengthM 1
CharsM
"y" "\0"
AR
Main
z
x[0]
x[1]
lengthM 5
CharsM
lengthM 1
CharsM
"E" "N" "E" "L" "y" "\0"
"0" "\0"
C
lengthM 0
CharsM
"\0" ?? ??
ar

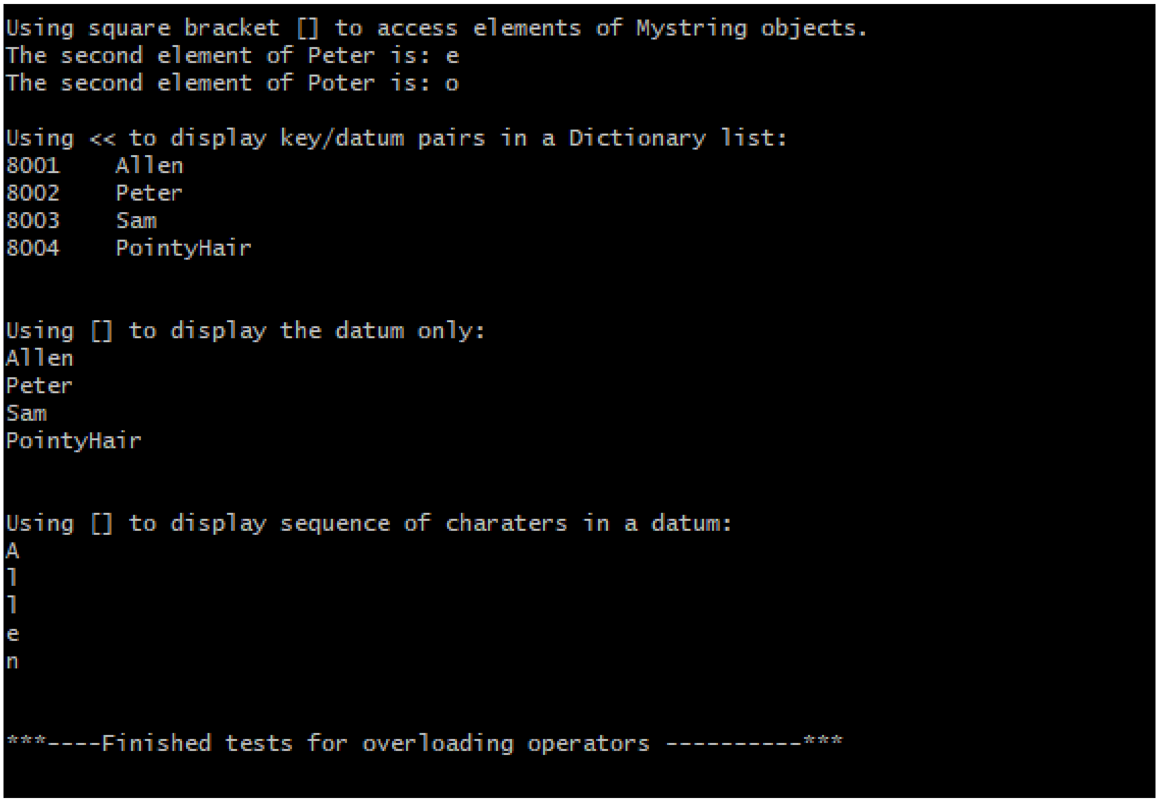
Exercise B Part I



Exercise B/C sample output



```
$ ./exB

Printing list just after its creation ...
  List is EMPTY.

Printing list after inserting 3 new keys ...
  8001  Dilbert
  8002  Alice
  8003  Wally

Printing list after removing two keys and inserting PointyHair ...
  8003  Wally
  8004  PointyHair

Printing list after changing data for one of the keys ...
  8003  Sam
  8004  PointyHair

Printing list after inserting 2 more keys ...
  8001  Allen
  8002  Peter
  8003  Sam
  8004  PointyHair
***----Finished dictionary tests---------------------------***

Printing list--keys should be 315, 319
  315  Shocks
  319  Randomness
Printing list--keys should be 315, 319, 335
  315  Shocks
  319  Randomness
  335  ParseErrors
Printing list--keys should be 315, 335
  315  Shocks
  335  ParseErrors
Printing list--keys should be 319, 335
  319  Randomness
  335  ParseErrors
Printing list--keys should be 315, 319, 335
  315  Shocks
  319  Randomness
  335  ParseErrors
***----Finished tests of copying----------------------***


Let's look up some names ...
  name for 8001 is: Allen.
  Sorry, I couldn't find 8000 in the list.
  name for 8002 is: Peter.
  name for 8004 is: PointyHair.
***----Finished tests of finding -------------------------***


Testing a few comparison and insertion operators.

Peter is greater than or equal Allen
Allen is less than Peter
Peter is not equal to Allen
Peter is greater than Allen
Peter is not less than Allen
Peter is not equal to Allen

Using square bracket [] to access elements of Mystring objects.
The second element of Peter is: e
The second element of Poter is: o
```



```
Using square bracket [] to access elements of Mystring objects.
The second element of Peter is: e
The second element of Poter is: o

Using << to display key/datum pairs in a Dictionary list:
8001    Allen
8002    Peter
8003    Sam
8004    PointyHair


Using [] to display the datum only:
Allen
Peter
Sam
PointyHair


Using [] to display sequence of charaters in a datum:
A
l
l
e
n


***----Finished tests for overloading operators ----------***
```