```
cat -b p4.cpp
     1	/*
     2	 PROGRAM NAME: Program 4: Random Access Files
     3	
     4	 PROGRAMMER:   James Francis
     5	
     6	 CLASS:        CSC 331.001, Fall 2014
     7	
     8	 INSTRUCTOR:   Dr. Robert Strader
     9	
    10	 DATE STARTED: October 9, 2014
    11	
    12	 DUE DATE:     October 16, 2014
    13	
    14	 PROGRAM PURPOSE:
    15	
    16	 1) Create a blank binary file
    17	 2) Read in "inventory" updates from prog4.dat to Item objects
    18	 3) Write the Item objects to the previously created binary file
    19	 4) Print to console the item record in the file going from one
    20	    item to the next based on their nextItem variable
    21	
    22	
    23	 VARIABLE DICTIONARY:
    24	
    25	 handler: fileHandler object that will handle records within the file
    26	 lines: integer that counts the number of records from the update file
    27	 record: Item object used to hold values from the update file
    28	 stockNum: integer representation of an item's stock number
    29	 description: string - brief description of the current record
    30	 qty: integer used to hold the count of a given stockNum
    31	 nextItem: integer used to hold the next stock number in "inventory"
    32	
    33	 ADTs: none
    34	
    35	 FILES USED: prog4.dat
    36	
    37	
    38	 SAMPLE INPUTS:
    39	
    40	 10 zidgits 17 -1
    41	 14 lidgits 2 7
    42	 6 gidgits 12 8
    43	 1 bidgits 25 3
    44	 16 widgits 9 10
```

```
45        7 midgits 0 2
46        3 didgits 11 6
47        5 tidgits 6 16
48        2 pidgits 7 5
49        8 kidgits 6 14

50
51        SAMPLE OUTPUTS: (to prog4out.dat)
52
53         1 bidgits  25
54         3 didgits  11
55         6 gidgits  12
56         8 kidgits   6
57        14 lidgits   2
58         7 midgits   0
59         2 pidgits   7
60         5 tidgits   6
61        16 widgits   9
62        10 zidgits  17
63
64        ---------------------------------------------------------------------*/

65       #include "Item.h"
66       #include "fileHandler.h"

67       int main(int argc, const char * argv[]){

68           fileHandler handler = fileHandler();
69           int lines = 0;
70
71           //following block of code creates the output file
72           ofstream openFile;
73           openFile.open("prog4out.dat", ios::out|ios::binary);
74           openFile.close();
75           //end creation of output file
76
77           fstream myFile("prog4out.dat", ios::in|ios::out|ios::binary);
78           Item record;
79
80           // files the output file with empty Item objects, each 20 bytes in size
81           // 21 items are written, the first is a placeholder for the header record
82           int i = 0;
83           for(i = 0; i<21;i++){
84               myFile.write(reinterpret_cast< const char * >(&record), sizeof(Item));
85           }
```

```
    86          fstream infile("../instr/prog4.dat", ios::in|ios::binary);
    87
    88          int stockNum;
    89          string description;
    90          int qty;
    91          int nextItem;

    92          while (infile >> stockNum >> description >> qty >> nextItem) {
    93
    94              //Creates Item objects and writes the information contained within the object to the output file
    95              record = Item(stockNum, description, qty, nextItem);

    96              //following line moves the filepointer to the item's location in bytes on the list
    97              myFile.seekg(record.getStockNum()*sizeof(Item));
    98
    99              //following line writes the information to the output file
   100              myFile.write(reinterpret_cast<const char *>(&record), sizeof(Item));

   101
   102              lines++;
   103          }

   104          infile.close();//close input file
   105
   106          myFile.seekg(0);//move filepointer to the beginning of the filestream
   107
   108          handler.updateHeader(lines, myFile);//update the header record with the number of records now in the file
   109
   110
   111          handler.printRecords(myFile);//print the records, sequentially, from the updated file
   112
   113          myFile.close();//close output file

   114          return 0;
   115      }


cat -b Item.h
     1      //
     2      //  Item.h
     3      //  p4
     4      //
     5      //  Created by James Francis II on 10/9/14.
     6      //  Copyright (c) 2014 James Francis II. All rights reserved.
     7      //
```

```
     8      #ifndef __p4__item__
     9      #define __p4__item__

    10      #include <stdio.h>
    11      #include <iostream>
    12      #include <cstdlib>
    13      #include <fstream>

    14      using namespace std;

    15      class Item{
    16      public:
    17          Item();
    18          Item(int stockNum, string description, int qty, int nextItem);
    19          int getStockNum() const;
    20          string getDescription() const;
    21          int getCount() const;
    22          int getNext() const;
    23
    24      private:
    25          int stockNum;
    26          char description[8];
    27          int count;
    28          int next;
    29          void setStockNum(int n);
    30          void setDescription(string str);
    31          void setCount(int n);
    32          void setNext(int n);
    33      };


    34      #endif /* defined(__p4__item__) */
:
printf \\n\\n


cat -b Item.cpp
     1      //
     2      //  Item.cpp
     3      //  p4
     4      //
     5      //  Created by James Francis II on 10/9/14.
     6      //  Copyright (c) 2014 James Francis II. All rights reserved.
     7      //
```

```
 8      #include "Item.h"
 9      Item::Item(){
10          //---------------------------------------------------------------
11          // Default Item Constructor
12          //---------------------------------------------------------------
13
14          setStockNum(0);
15          setDescription("");
16          setCount(0);
17          setNext(0);
18      }

19      Item::Item(int stock, string desc, int qty, int nextItem){
20          //---------------------------------------------------------------------
21          //Preconditions: 3 integers and a string are passed to the constructor
22          //               from the calling code
23          //Postconditions: an Item object is instantiated with the passed values
24          //
25          //Variables used: stock - integger representing an item's stock number
26          //                 desc - description of an item, in a string
27          //                  qty - integer value with the current quantity
28          //             nextItem - next item in the stock list
29          //---------------------------------------------------------------------

30          setStockNum(stock);
31          setDescription(desc);
32          setCount(qty);
33          setNext(nextItem);
34      }

35      //---------------------------------------------------------------
36      //BEGIN GETTERS AND SETTERS
37      //---------------------------------------------------------------

38       int Item::getStockNum()const{
39          return stockNum;
40      }

41       string Item::getDescription()const{
42          return description;
43      }

44       int Item::getCount()const{
45          return count;
```

```
    46      }

    47       int Item::getNext()const{
    48          return next;
    49      }

    50      void Item::setStockNum(int stock){
    51          stockNum = stock;
    52      }

    53      void Item::setDescription(string desc){
    54          for (int i = 0; i<8;i++){
    55              description[i]=desc[i];
    56          }
    57
    58      }

    59      void Item::setCount(int qty){
    60          count = qty;
    61      }

    62      void Item::setNext(int nextItem){
    63          next = nextItem;
    64      }

    65      //------------------------------------------------------------------
    66      //END GETTERS AND SETTERS
    67      //------------------------------------------------------------------
:
printf \\n\\n


cat -b fileHandler.h
     1      //
     2      //  fileHandlerTest.h
     3      //  p4
     4      //
     5      //  Created by James Francis II on 10/13/14.
     6      //  Copyright (c) 2014 James Francis II. All rights reserved.
     7      //

     8      #ifndef __p4__fileHandlerTest__
     9      #define __p4__fileHandlerTest__

    10      #include <stdio.h>
```

```
    11      #include <iomanip>
    12      #include <fstream>

    13      class fileHandler{
    14      public:
    15
    16          fileHandler();
    17          void outputLine(ostream &output, const Item & );
    18          void updateHeader(int lines, fstream& myFile);
    19          void printRecords(fstream& myFile);
    20
    21      private:
    22
    23
    24      };

    25      #endif /* defined(__p4__fileHandlerTest__) */
printf \\n\\n


cat -b fileHandler.cpp
     1      //
     2      //  main.cpp
     3      //  p4
     4      //
     5      //  Created by James Francis II on 10/7/14.
     6      //  Copyright (c) 2014 James Francis II. All rights reserved.
     7      //

     8      #include "Item.h"
     9      #include "fileHandler.h"


    10      fileHandler::fileHandler() {
    11          //---------------------------------------------------------------
    12          //DEFAULT FILEHANDLER CONSTRUCTOR
    13          //---------------------------------------------------------------
    14          }

    15      void fileHandler::outputLine(ostream &output, const Item &record){
    16          //---------------------------------------------------------------
    17          //Preconditions: a reference to an output stream and a reference
    18          //               to an Item object were passed by the calling code
    19          //Postconditions: the contents of the record object are sent to
    20          //                the output stream
```

```
21          //
22          //Variables used: &output: reference to an output stream
23          //                 &record reference to an Item object
24          //-----------------------------------------------------------
25          output << right << setw(4)<< record.getStockNum()<<right <<setw(8) << record.getDescription() <<
right<<setw(4)<< record.getCount();

26          cout.clear();

27
28      }

29      void fileHandler::updateHeader(int lines, fstream& myFile){
30          //-----------------------------------------------------------
31          //Preconditions: a reference to a file stream and an integer
32          //               were passed by the calling code.
33          //Postconditions: An integer value representing the number of
34          //                valid records is written to the Header record
35          //
36          //Variables used: lines -  integer representing valid records
37          //               myFile - reference to an fstream output file
38          //-----------------------------------------------------------
39
40          myFile.seekp(0);
41          int stockNum = lines;
42          string description = "Records";
43          int qty = 0;
44          int nextItem = 0;
45          Item record = Item(stockNum, description, qty, nextItem);
46
47          // following line writes the information stored in the object record
48          // to the output filestream
49          myFile.write(reinterpret_cast<const char *>(&record), sizeof(Item));
50          myFile.seekp(0);
51      }

52      void fileHandler::printRecords(fstream& myFile){
53          //-----------------------------------------------------------------------
54          //Preconditions: a reference to a file stream is passed from the calling code.
55          //Postconditions: All records are printed in a random access fashion
56          //
57          //Variables used: lines -  integer representing valid records
58          //               myFile - reference to an fstream output file
59          //-----------------------------------------------------------------------
60
```

```
    61
    62          int n=0;
    63          Item record;
    64          bool eof = false;
    65          cout<<endl;
    66          while (eof==false) {
    67              myFile.seekg(n*sizeof(Item));

    68              myFile.read(reinterpret_cast<char * >(&record), sizeof(Item));
    69
    70              if (record.getStockNum()>0 && record.getNext()>0  ) {
    71
    72                  // This statement sends the record to be printed and cout
    73                  // so the record's content can be printed to console
    74                  // n is set to the next item so the pointer within the fstream
    75                  // can be moved to the next item in sequence
    76
    77                  outputLine(cout, record);
    78                  cout<<endl;
    79                  cout.clear();
    80                  n=record.getNext();

    81              }else if(record.getStockNum()>0 && record.getNext()==-1){
    82
    83                  // This statement sends the last Item to be printed
    84                  // so the record's content can be returned to console
    85                  // eof is encountered because a nextItem value of -1
    86                  // indicates the last Item
    87
    88                  outputLine(cout, record);
    89                  cout<<endl;

    90                  cout.clear();
    91                  eof = true;
    92
    93              } else n=1;
    94
    95          }
    96          cout<<endl;
    97      }
printf \\n\\n


:
```

```
g++ fileHandler.cpp Item.cpp  p4.cpp -o p4
:
p4

   1 bidgits  25
   3 didgits  11
   6 gidgits  12
   8 kidgits   6
  14 lidgits   2
   7 midgits   0
   2 pidgits   7
   5 tidgits   5
  16 widgits   9
  10 zidgits  17
```