CSE 373 Homework 5 Project Description

Assigned: Wednesday, May 8, 2002
Due: Wednesday, May 15, 2002
At the start of class

Introduction

For this homework project, you will develop a code module that implements a binary heap package and answer some questions. All the source files that we are providing for this assignment are in a zip file on the web site. You should download and unzip the file. The questions were handed out in class; you can also get a copy from the web site.

mainBHeap. The mainBHeap program calls the procedures in your bheap.c package to create a binary heap (priority queue). It adds and deletes Symbol objects, and prints a description of the heap that is suitable for processing with dot to build a graphic display of the heap. As you remember from previous homework, dot is a program that takes text input files describing a graph of some sort, and produces postscript output files with nicely drawn plots. The drawing in this case is a drawing of the binary heap that your procedures have created.

The main program for the homework is supplied in ADT/BHeap/mainBHeap.c, and you are to write the individual heap management functions as defined in ADT/include/bheap.h. The functions are based very closely on the discussion in the textbook, however, they are not exactly the same.

Grading

The 7 homework assignments of the quarter will count for a total of 50% of your class grade, and each individual homework assignment will count for about 7% of the total class grade. The grading for this project is as follows.

Questions:	10 points
Implementation:	10 points
Total	20 points

NOTE: You need to turn in several things:

1. the paper copy of your answer sheet
2. a printout of a winpidplot.dot (or pidplot.dot) showing the output from your program
3. a printout of the plot generated by dot from the file in item 2
4. a copy of the receipt you got when you did the web turn in
5. and do a web turnin of your implementation of bheap.c

Staple the answer sheet, the dot file, the plot, and the receipt together and turn them in on Wednesday.

Directory Structure

All of the homework projects are implementations of one of the Abstract Data Types that we discuss in class. The directory structure of the files you receive is as follows:

ADT/	top level directory
ADT/include	header files
ADT/symbols	example symbol table files
ADT/BHeap	directory for the BHeap project
ADT/Lecture	examples from the lectures, if any
ADT/lib	precompiled binaries, if any

Program: mainBHeap

The purpose of this program is to read a symbol table file and create a min-ordered binary heap from it, then create a drawing showing the structure of the heap.

The program creates a min-heap sorted by symbol name. The program prints information about the resulting heap in the format defined by the dot program. Input is taken from stdin or a named symbol table file, output is to stdout (which can be redirected to a disk file). The dot program reads this file and produces a postscript drawing. After creating the heap, the main program deletes elements one after another until the heap is empty.

This program requires you to implement a small set of heap management functions. The main program and the header files are supplied; you add code to the implementation file "bheap.c" to supply the actual heap management functions.

Program: dot

This program is already written (by ATT). It is installed on the lab computers, and it is available from the class web site for installing on your own system. You don't need to know anything about how the input to dot is formatted in order to do this assignment, since the format routines are already written and supplied to you in the homework zip file. If you are running on your own system, you will have to do the installation of dot (which is just installing the graphviz download and making it available in your path).

If you want to add printf statements to your code, you should write out your information surrounded by /* and */, because then dot can still read the output file and it will ignore these lines (it considers them to be comments). See mainBHeap.c for an example of how to do this (the name of the Element deleted by DeleteMin is printed by mainBHeap.c).

When I print the plots on my Win2K system using GSView for Windows, I use the following settings: Media->Rotate Media, Media->Letter, Orientation->Portrait, and then I tell the printer to print in Landscape mode. This prints the complete tree diagram with no clipping.

To do:

1. Create the BHeap project just as you have done for the previous projects. For the Program arguments entry, use "..\symbols\pidsymbols.txt".
2. Note that although there is a file "bheap.c" provided to you it is very incomplete, and so the project will not link correctly. All the routines that you will implement are missing.
3. Change the name that is included in the function getBHeapAuthor. As delivered to you, it says "Anonymous Author." You should change that to be your own name.
4. Write the routines that are needed by mainBHeap as described below, rebuild the project, and run it. Debug until done.
5. If you run the project from Visual C, the output gets displayed in a console window. This may not be very understandable. I have supplied a batch file called winrunit.bat that runs your program (Debug\bheap.exe) from a command line and stores the output in winpidplot.dot. It then runs dot, and stores the result in winpidplot.ps. To use this batch file, get a command prompt (Start->Programs->Accessories->Command Prompt) and use the cd command to move to the ADT\BHeap directory. Then type "winrunit".

   The winpidplot.dot file contains whatever information your program wrote out. If you have been careful to surround any output you have added with /* and */ then dot should run okay.

   The winpidplot.ps file is a postscript file that contains an actual drawing of the binary heap that was created. You can use Ghostview (available from our web site) or any other postscript viewer to display and print the drawings.

6. Review the code as needed in order to answer the questions in the homework.

The definition of struct HeapStruct is in include/privatebheap.h. It is not in bheap.c because the plot functions need to be able to see the structure definition also. Similarly, the definition of struct Symbol is in include/symbol.h because the symbol functions in symbol.c need to be able to see the structure of a Symbol.

The function headers for the routines you need to write are in ADT/include/bheap.h. Note that the functions are very similar, but NOT IDENTICAL, to the ones defined in the textbook.

The functions are as follows.

char *getBHeapAuthor(void)

Returns a text string to the caller naming the person who wrote the program. Change "Anonymous Author" to your name.

PriorityQueue CreatePriorityQueue(int Cap,ElementType MIN_E, Comparator C);

Create a new Priority Queue. This is the Initialize procedure from the book. You need to store the pointer to the minimum element and the pointer to the comparison function for later use.

void DestroyPriorityQueue(PriorityQueue H);

Release all the memory allocated in CreatePriorityQueue. As in all previous projects, this package is only responsible for the memory it has allocated. Releasing the memory allocated by the caller is the caller's responsibility.

void InsertElement(PriorityQueue H, ElementType X);

Add an element to the binary heap and reorganize to maintain the min-heap order property. If the heap is full, the program can terminate immediately using the FatalErrorBound macro, if you like.

ElementType DeleteMinElement(PriorityQueue H);

Remove the minimum element from the binary heap and reorganize to maintain the min-heap order property. If the queue is empty, return NULL.

int IsEmpty(PriorityQueue H);

Return TRUE if the heap is empty, FALSE if it is not empty. You can use 1 and 0 for these values.

int IsFull(PriorityQueue H);

Return TRUE if the heap is full, FALSE if it is not full. You can use 1 and 0 for these values.