CS5521

Assignment 4

Submitted by : FERDOUS AHMED

// binheap.cpp implementation file for binary heaps

#include "binheap.h"

#include <iostream>

#include <iomanip>

// Utility routines used to implement binary heaps and heapSort

int parent ( int i ) { return i/2 ; }

int left ( int i ) { return 2\*i ; }

int right ( int i ) { return 2\*i + 1 ; }

void BinHeap :: minHeapify ( int i, int \*A)

{

int l, r, smallestloc ;

KeyType temp ;

l = left( i ) ;

r = right( i ) ;

// You write the rest of this - remember this is a \_min\_ heap

if (l<heapSize && A[l]<A[i])

{

smallestloc = l;

}

else{

smallestloc = i;

}

if (r<heapSize && A[r]<A[smallestloc])

{

smallestloc=r;

}

if (smallestloc!= i){

temp=0;

temp= A[smallestloc];

A[smallestloc]=A[i];

A[i]=temp;

minHeapify(smallestloc,A);

}

}

void BinHeap :: buildMinHeap( )

{

// You implement this for heapSort in Assignment # 5

}

// Constructors, Destructor

BinHeap :: BinHeap() {

heapSize = 0 ; length = 1000 ; A = new KeyType[1001] ;

}

BinHeap :: BinHeap( int arrayLength ) {

heapSize = 0 ; length = arrayLength ; A = new KeyType[arrayLength+1] ;

}

BinHeap :: ~BinHeap() { delete[] A ; }

// Min-heap, min priority queue operations

void BinHeap :: insert( KeyType key )

{

heapSize++ ;

A[heapSize] = key ; // <-- This seems to work.

decreaseKey( heapSize, key ) ;

}

KeyType BinHeap :: minimum( )

{

return A[1] ; // Assumes heap is not empty!

}

KeyType BinHeap :: extractMin( )

{

KeyType min ;

if ( heapSize < 1 ) {

cout << "\nHeap underflow in extractMin()\n\n" ; return 0 ;

}

// You write the rest of this - remember this is a \_min\_ heap

min = A[1] ;

A[1] = A[heapSize];

heapSize = heapSize-1;

minHeapify (1,A);

return min ;

}

void BinHeap :: decreaseKey( int i, KeyType key )

{

KeyType temp ;

if ( key > A[i] )

{

cout << "\nKey larger than A[i] in decreaseKey()\n\n" ; return ;

}

A[i] = key;

while (i > 1 && A[parent(i)]> A[i])

{

//swap(A[parent(i)],A[i]);

KeyType Temp = A[i];

A[i] = A[parent(i)];

A[parent(i)]=Temp;

i = parent(i);

// You write the rest of this - remember this is a \_min\_ heap

}

}

// Auxiliary operations

// Shows the structure of the binary heap

void BinHeap :: ShowHeap( int i, int depth )

{

if ( i <= heapSize ) {

ShowHeap( right(i), depth+1 ) ;

cout << setw( depth\*6 +4 ) << A[i] << endl ;

ShowHeap( left(i), depth+1 ) ;

}

}

// Just prints A[] as a horizontal array; may be used for debugging

void BinHeap :: PrintHeap()

{

cout << "Heap elements in the array:\n" ;

for ( int i = 1 ; i <= heapSize ; i++ )

cout << A[i] << " " ;

cout << endl ;

}

// A routine that is useful for testing decreaseKey()

int BinHeap :: search( KeyType key )

{

for ( int i = 1 ; i <= heapSize ; i++ )

if ( A[i] == key ) return i ;

return 0 ;

}

// For Assignment # 5

void BinHeap :: minHeapSort( )

{

KeyType temp ; // Used for swapping

// You implement this for Assignment # 5

}

**Heaptest1 Output**

# create empty heap (this is already done in the command interpreter)

# insert 4 items, printing the heap after each insert

Structure of heap (rotated 90 degrees to left):

491

Structure of heap (rotated 90 degrees to left):

122

491

Structure of heap (rotated 90 degrees to left):

245

122

491

Structure of heap (rotated 90 degrees to left):

245

122

474

491

# The following command tests minimum()

Minimum = 122

# The following command tests decreaseKey(), decreaseing 491 to 111

Structure of heap (rotated 90 degrees to left):

245

111

122

474

# extract mins (quitely after 1st) until empty, printing heap after each one

Minimum extracted = 111

Structure of heap (rotated 90 degrees to left):

245

122

474

Structure of heap (rotated 90 degrees to left):

245

474

Structure of heap (rotated 90 degrees to left):

474

Structure of heap (rotated 90 degrees to left):

# attempt to extract min from empty heap

Heap empty, can't extract minimum.

# end of test

**Heaptest2 Output**

# create empty heap (this is already done in the command interpreter)

# insert 20 keys

# show heap structure

Structure of heap (rotated 90 degrees to left):

15

7

14

3

13

6

12

1

11

5

10

20

2

19

9

18

4

17

8

16

# do 10 extract mins with print key flag off

# show heap structure

Structure of heap (rotated 90 degrees to left):

14

12

13

11

18

19

15

17

16

20

# do 10 extract mins with print key flag on

Minimum extracted = 11

Minimum extracted = 12

Minimum extracted = 13

Minimum extracted = 14

Minimum extracted = 15

Minimum extracted = 16

Minimum extracted = 17

Minimum extracted = 18

Minimum extracted = 19

Minimum extracted = 20

# show heap structure (should be empty)

Structure of heap (rotated 90 degrees to left):

# do an extract min from empty heap

Heap empty, can't extract minimum.

# quit

**Heaptest3 Output**

# create empty heap (this is already done in the command interpreter)

# insert 50 items

# print the entire heap

Structure of heap (rotated 90 degrees to left):

99

96

304

88

266

236

372

49

387

193

457

122

302

406

272

449

390

491

24

366

224

405

202

437

218

368

90

363

181

355

177

299

245

440

72

149

145

298

100

291

228

473

74

290

201

438

199

403

399

474

# do 459 alternating extract mins and inserts quietly

# do 40 extract mins with print flag on

Minimum extracted = 315

Minimum extracted = 460

Minimum extracted = 449

Minimum extracted = 461

Minimum extracted = 462

Minimum extracted = 463

Minimum extracted = 464

Minimum extracted = 465

Minimum extracted = 466

Minimum extracted = 467

Minimum extracted = 468

Minimum extracted = 469

Minimum extracted = 470

Minimum extracted = 471

Minimum extracted = 472

Minimum extracted = 473

Minimum extracted = 474

Minimum extracted = 475

Minimum extracted = 476

Minimum extracted = 477

Minimum extracted = 478

Minimum extracted = 479

Minimum extracted = 480

Minimum extracted = 481

Minimum extracted = 482

Minimum extracted = 483

Minimum extracted = 484

Minimum extracted = 485

Minimum extracted = 486

Minimum extracted = 487

Minimum extracted = 488

Minimum extracted = 489

Minimum extracted = 490

Minimum extracted = 491

Minimum extracted = 492

Minimum extracted = 493

Minimum extracted = 494

Minimum extracted = 495

Minimum extracted = 496

Minimum extracted = 497

# print the remaining heap

Structure of heap (rotated 90 degrees to left):

500

499

504

498

503

506

501

507

502

505

# extract the rest

Minimum extracted = 498

Minimum extracted = 499

Minimum extracted = 500

Minimum extracted = 501

Minimum extracted = 502

Minimum extracted = 503

Minimum extracted = 504

Minimum extracted = 505

Minimum extracted = 506

Minimum extracted = 507

# attempt to extract min from empty heap

Heap empty, can't extract minimum.

# ene of test