Lecture 14.1

Topics:

1. Heap Initialization – Heapification

---

1. Heap Initialization – Heapification

The heapification can be done by working with one value at a time and then loop through all parent nodes to complete the heapification.

```
/**
 * Program Name:    cis27Lec1411Heapification.c
 * Discussion:      Max Heap
 */

#include <stdio.h>

/**
  Assuming that an array of known size is given. All data
  values are stored in the array from index iLow to an index
  of iHigh.

  The heapifyOne() function will work on one element (at the
  root) at a time.

  The heapify() function will work on the array to convert
  it to a max heap. This max heap will start at index 0
  and has ( iHigh - iLow + 1 ) elements.

  heapify() will loop through the calls of heapifyOne() to
  complete the rearrangement.

  The heap is stored right back to the given array iAry[].
 */

void heapifyOne(int iAry[], int iLow, int iHigh) {
  int iIndex;

  int iTemp = iAry[iLow]; /* copying of root node */

  iIndex = 2 * iLow + 1;    /* index of left child */

  while (iIndex <= iHigh) {
    if (iIndex < iHigh) {
      if (iAry[iIndex] < iAry[iIndex + 1]) {
        iIndex = iIndex + 1; /* index of the largest child */
      }
    }

    if (iTemp > iAry[iIndex]) { /* subtree is already a heap */
      break;
    } else {
      iAry[iLow] = iAry[iIndex]; /*moving larger child to root*/
      iLow = iIndex; /*going to subtree to restore heap*/
      iIndex = 2 * iLow + 1;
```

```
    }
  }

  iAry[iLow] = iTemp;

  return;
}

void heapify(int iAry[], int iLow, int iHigh) {
  int i;
  int iLength;

  iLength = iHigh - iLow + 1;

  for (i = iLength / 2 - 1; i >= 0; i--) {
    heapifyOne(iAry, i, iLength - 1);
  }

  return;
}


int main() {
  int ch;

  int iAry[] = {20, 12, 35, 15, 10, 80, 30, 17, 2, 1};

  int iDataSize = 10;
  int i;

  printf("Original Array: \n");
  for (i = 0; i < iDataSize; i++) {
    printf("\niAry[ %d ] : %d", i, iAry [i]);
  }

  heapify(iAry, 0, 9);

  printf("\n\nAfter Heapifying: \n");
  for (i = 0; i < iDataSize; i++) {
    printf("\niAry[ %d ] : %d",i, iAry[i]);
  }

  printf("\nEnter a character + ENTER to stop ...\n");
  scanf("%d", &ch);

  return 0;
}

/* PROGRAM OUTPUT
Original Array:

iAry[ 0 ] : 20
iAry[ 1 ] : 12
iAry[ 2 ] : 35
iAry[ 3 ] : 15
iAry[ 4 ] : 10
iAry[ 5 ] : 80
iAry[ 6 ] : 30
iAry[ 7 ] : 17
```

```
iAry[ 8 ] : 2
iAry[ 9 ] : 1

After Heapifying:

iAry[ 0 ] : 80
iAry[ 1 ] : 17
iAry[ 2 ] : 35
iAry[ 3 ] : 15
iAry[ 4 ] : 10
iAry[ 5 ] : 20
iAry[ 6 ] : 30
iAry[ 7 ] : 12
iAry[ 8 ] : 2
iAry[ 9 ] : 1
Enter a character + ENTER to stop ...
q
*/
```