BinaryHeap Chocolate Code-off

Problem 1: Identify which of the following is/are binary heaps:

A.__________

0	1	2	3	4	5	6	7	8	9
stuffing	peppermint	orange	turkey	humming	holiday	casserole	autumn	dates	apples

B.___________

0	1	2	3	4	5	6	7	8	9
42	27	15	32	73	29	94	65	26	99

C. ___________

0	1	2	3	4	5	6	7	8	9
42	28	40	18	24	39	36	8	`14	23

D. ___________

0	1	2	3	4	5	6	7	8	9	10
vacations	let	students	fly	home	so	relaxing	and	`eating	can	happen

Problem 2: Fill in the blank so that the bubbleUp method works:

```
void BinHeap::bubbleUp(int i) {
       int parent = ______________;
       if ((parent >-1)  && (heap[parent] < heap[i] )) {
              string tmp = heap[i];
              heap[i] = heap[parent];
              heap[parent] = tmp;
              bubbleUp(parent);
       }
}
```

Problem 2b: Is this for an insertion or a deletion? _________________________________

Problem 3: If it takes in the worst case 4 comparisons to bubble up, AT MOST, how many nodes are in the heap? ___________

Problem 3: Given the following array, show the array after inserting 27 into the heap:

0	1	2	3	4	5	6	7	8
32	17	24	12	15	20	19	2	7

Problem 4: Given the following heap, perform a delete:

0	1	2	3	4	5	6	7	8	9
Travel	Suitcase	Passenger	Starbucks	Driving	Homeward	Gps	Ride	Departure	Coffee

Problem 5: Use the optimal method to make a heap out of the following:

0	1	2	3	4	5	6	7	8	9
Carrot	And	Suggest	I	Cake	Pumpkin	Pie	Vegetables	Bean	brownies

Problem 6: Show a heapsort on the following heap:

0	1	2	3	4	5	6	7	8
y	s	o	l	p	g	e	i	a

Problem 7: Show the steps in finding the 5th LARGEST element in the most efficient way, showing steps (Note: in this array, the 5th largest would be: tofurkey)

gobble tofurkey wishbone gravy apples cranberries harvest yams turnips veggies baking fireplace gourd

Problem 8: Given the following doubly linked list, what is created? ______________________

"c"<->"h"<->"a"<->"b"<->"o"<->"n"<->"c"<->"t"<->"i"<->"d"<->"k"<->"p"<->"s"<->"n"<->"y"<->"e"

```
void LL::sillyfunc2() {
       int i = 0;
       int j = 2;
       Node *tmp = first; // the first element in the list
       while (tmp!= NULL) {
              Node *tmp2 = tmp;
              int k = 0;
              while (k < j && tmp2!= NULL) {
                     tmp2 = tmp2->next;
                     k++;
              }
              if (tmp2!= NULL) {
                     tmp->next = tmp2;
                     tmp2->prev = tmp;
              }
              else {
                     tmp->next = NULL;
                     last = tmp;
                     size = i;
              }
              tmp = tmp2;
              j++;
              i++;
       }
}
```