Worksheet 6

Overview

The w orkshop for Week 7 w ill start w ith a tutorial on Quicksort, and you w ill implement a Hash Function / Hash Table.

Tutorial Questions

Question 6.1 You are asked to show the operation of quicksort on the follow ing keys. For simplicity, use the rightmost element as the partition element:

- 2 3 97 23 15 21 4 23 29 37 5 23
  Comment on the stability of quicksort and its behaviour on almost sorted inputs.

Programming Exercises

Programming 6.1 Implementing a hash table w ill help you understand how it w orks. In this laboratory, you are asked to implement an open addressing hash table that uses linear probing for collision resolution. In the interests of simplicity, and to help you implement the hash table quickly, you can simply store integers (keys), keep the table very small (suggested size 13, w hy?), and use a very simple hash function, e.g. `f(key) = (key*97)%13`. Implement a function *printTable(ht)*, so that you can check w hat is in the hash table at every step. Remember that you w ill get a segmentation fault if you try to print a *NULL* value, so w rap the printing w ithin a conditional statement that test for *NULL*.

Because the table is small, you can engineer collisions. Check w hat happens w hen you get a collision, then w hen you get tw o collisions, etc. Think about how you could handle the situation w here the table is almost full - How nearly full are you going to let it get? Could you expand the table? If you expand the table, w hat w ill happen to the items that are already in the table?

Programming 5.2's material may be useful for reference if you get stuck, though it does not handle unresolvable collisions w ell. Likew ise, not w orrying about polymorphism may save you some time and confusion.

```
#include <stdio.h>
#include <stdlib.h>

#define INPUTSIZE 14

/* Declare hash table data structure */

/* Write insert function */

/* Write printTable function. */

/* Write freeTable function. */

int main(int argc, char **argv){

    int inputs[INPUTSIZE] = {2, 3, 97, 23, 15, 21, 4, 23, 29, 37, 5, 23, 28, 40};

    /* Declare table. */

    int i;

    /* Create empty table. */
    printf("Created hash table of size 13\n");
    printTable(/* table */);

    for(i = 0; i < INPUTSIZE; i++){
        printf("Inserting %d\n",inputs[i]);
        insert(/* table */, inputs[i]);
        printTable(/* table */);
    }
    printf("Finished inserting items into table\n");

    freeTable(/* Table */);

    return 0;
}
```