COP 3502 Section 4 Exam 2A – Sorting, Tries, Hash Tables, Base Conversion Solutions (Thursday 11/16/2023)

1) (5 pts) Show the state of the array below after each iteration of Selection Sort. (Note: you are selecting for the maximum item in each iteration.) The result of the last iteration has been provided for you.

Initial Values	11	6	8	13	9	1	4
1st iteration	11	6	8	4	9	1	13
2nd iteration	1	6	8	4	9	11	13
3rd iteration	1	6	8	4	9	11	13
4th iteration	1	6	4	8	9	11	13
5th iteration	1	4	6	8	9	11	13
6th iteration	1	4	6	8	9	11	13

Grading: 1 pt for each row, row must be completely correct to get the point.

2) (5 pts) Both Quick Sort and Merge Sort can be sped up by making a base case of roughly n = 40 and then running insertion sort on the subsection of the array from index low to index high, inclusive. Complete the code below so that it performs an Insertion Sort on the subsection of array starting at index low and ending at index high, inclusive.

```
void insertionsort(int* array, int low, int high) {

    for (int i=low+1; i<=high; i++) {

        int tmpIdx = i;

        while (tmpIdx > low && array[tmpIdx] < array[tmpIdx-1]) {

            int tmp = array[ tmpIdx ];

            array[ tmpIdx ] = array[ tmpIdx-1 ];

            array[tmpIdx-1] =  tmp;

            tmpIdx-- ;
        }

    }
}
```

Grading: 1 pt for each slot

3) (10 pts) Consider the problem of printing out all words in a trie that have unique letters only (no repeats). We can solve this problem by writing a recursive function that takes in the "current" node to the trie, as well as a boolean array of size 26 (array of int but just storing 0 or 1), marking which letters have already been "used" to traverse down to this node in the trie, along with a string, word, storing the word formed so far, and an integer, k, representing how far down the trie we are. Complete the function below to accomplish this task.

```
typedef struct trie {
    int isWord;
    struct trie* next[26];
} trie;

void printunique(trie* root, char word[], int k, int used[]) {

    if (root == NULL) return;

    if (root->isWord) {

        word[k] = '\0' ;            // 1 pt

        printf("%s\n", word);
    }

    for (int i=0; i<26; i++) {

        if (used[i]) continue ;       // 1 pt

        used[i] = 1 ;                                  // 1 pt

        word[k] = (char)('a'+i);                       // 2 pts

        printUnique(root->next[i], word, k+1, used) ; // 4 pts

        used[i] = 0  ;                                 // 1 pt
    }

}
```

Grading Note: for 2 pts cast to char not needed, just 'a'+i, do 1 pt each side of eqn
for 4 pts: 1 pt rec call, 1 pt 1st param, 1 pt k+1, 1 pt rest

4) (9 pts) Consider implementing a hash table storing integers using the hash function shown below (value is the input to the hash function and n is the size of the hash table the integers are being stored in) and the technique of linear probing to determine collisions.

```
int hashfunction2(int value, int n) {
    int res = 1;
    while (value > 0) {
        res = (res * (value%10))%n;
        value = value/10;
    }
    return res;
}
```

Using a hash table of size n = 13, if we were to insert the following integers, in the order shown below, determine which indexes each number would get inserted into. (Please draw the number in the appropriate slot in the array shown below. The array is shown over two lines so that there's enough room to write each number.)

Numbers to Insert: 234, 69, 34, 525, 72, 2323, 33, 117511, and 22222.

Index	0	1	2	3	4	5	6
Value	525	72	69	117511			22222

Index	7	8	9	10	11	12
Value			33	2323	234	34

Grading: 1 pt per slot (to get the point the correct number must be in the correct slot and no other number can be in the slot)

5) (10 Pts) Perform the requested base conversions. Note: only 1 pt is given for the answers. The rest of the points are awarded for showing the proper work (using the algorithms shown in class for each conversion.)

(a) (2 pts) 75_{10} = **1001011** $_2$ (convert 75 in base 10 to base 2)

Grading: 1 pt answer, 1 pt process (no guess and check)

2 | 75
2 | 37 R 1
2 | 18 R 1
2 | 9 R 0
2 | 4 R 1
2 | 2 R 0
2 | 1 R 0

(b) (2 pts) 3021_5 = **386** $_{10}$ (convert 3021 in base 5 to base 10)

Ans = $3 \times 5^3 + 0 \times 5^2 + 2 \times 5^1 + 1 \times 5^0 = 375 + 10 + 1 = 386$

Grading: 1 pt answer, 1 pt process

(c) (3 pts) 462107_8 = **26447** $_{16}$ (462107 in base 8 converted to base 16)

$462107_8 = 100110010001000111_2 = 10\ 0110\ 0100\ 0100\ 0111_2 = 26447_{16}$

Grading: 1 pt to binary, 1 pt grouping by 4s, 1 pt final answer
Give full credit if use base 10 as intermediary and they get the correct answer.

(d) (3 pts) $C24FE_{16}$ = **3022376** $_8$ (C24FE in base 16 converted to base 8)

$C24FE_{16} = 11000010010011111110_2 = 11\ 000\ 010\ 010\ 011\ 111\ 110_2 = 3022376_8$

Grading: 1 pt to binary, 1 pt grouping by 3s, 1 pt final answer
Give full credit if use base 10 as intermediary and they get the correct answer.

6) (1 pt) What fruit is used to produce the juice, Simply Orange? **Orange (Give to all)**