CS210 Data Structures
(202) Final Exam

Name:_____ ID_____

Check your section:

☐ Dr. Sawsan Alhalawani on Sunday	☐ Dr. Basit Qureshi
☐ Dr. Sawsan Alhalawani on Monday	☐ Dr. Muhammad Akour

Instructions:

- This exam contains four questions with multiple parts, on 5 sheets of papers.
- Time allowed: 120 minutes
- Closed Book, Closed Notes.
- Use of Calculators is ALLOWED. Use of other computing devices / smartphones etc is strictly prohibited.
- Answer the problems on the exam sheets only. No additional attachments would be accepted.
- When the "time is over" is called, it is the students' responsibility to submit his exam to the invigilator. Submitting completed exam 3 minutes after the "time is over" will incur a penalty of **5 points**.

Few gentle reminders:

- If you get stuck on some problem for a long time, move on to the next one.
- You should be better off by first reading all questions and answering them in the order of what you think is the easiest to the hardest problem.
- Keep the points distribution in mind when deciding how much time to spend on each problem.

Question No.	Part a	Part b	Part c	Part d	Student's Score
Question 1 (CLO 4)	/6	/2	/2		/10
Question 2 (CLO 3)	/2	/2	/3	/3	/10
Question 3 (CLO 2)	/5	/5			/10
Question 4 (CLO 1)	/4	/4	/2		/10
Total					/40

Question 1. [6 + 2 + 2 = 10 points]

Part a. [6 points] Ahmed wants to store some integers in a hash-table T1 of <u>size 11</u>. He wants to try out some hash functions to see if they would ensure uniform distribution of the data. The following functions are available f(x), g(x) and h(x).

$f(x) = x\%7$, $g(x) = (x * 2)\%11$, $h(x) = (x\%6) * 2$.

Insert the following data in the hash table: 3, 15, 17, 22, 19, 16, 27, 35, 46, 8.

Assume that collisions are resolved using linear probing method; compute the number of collisions and probes/displacement for each.

0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	

0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	

0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	

Number of collisions with f(x) _____

Number of collisions with g(x) _____

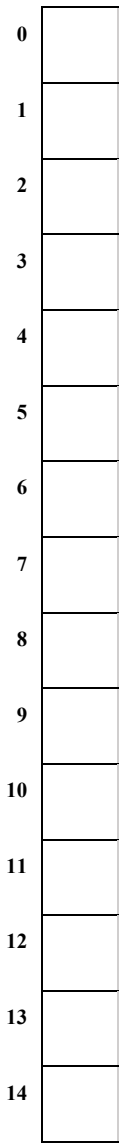
Number of collisions with h(x) _____

Number of probes/displacements with f(x) _____

Number of probes/displacements with g(x) _____

Number of probes/displacements with h(x) _____

Part b. [2 points] For the information provided in part(a); select the best hash function. Using this hash function, insert the same data (as in part a) into the hash tables T2 of size 15.



Number of collisions in T2

_____

Number of probes/displacements in T2

_____

Part c. [2 points] Comparing the Hash-tables T1 (part-a) and T2 (part-b); we observe fewer (or) more collisions and probes/displacements. Explain why?

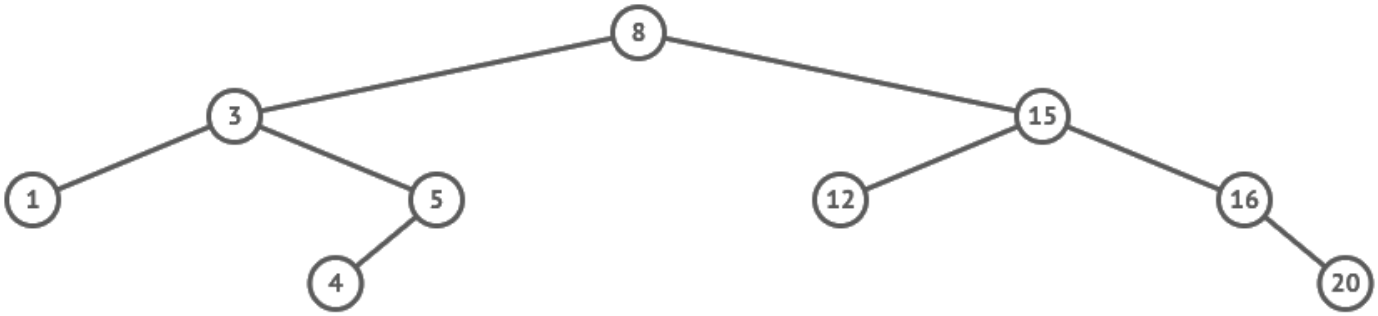
Question 2. [2 + 2 + 3 + 3= 10 points]

Part a. [2 points] Assume an AVL Tree T that stores integers as keys, is given. Write a method that returns the largest value in the tree.

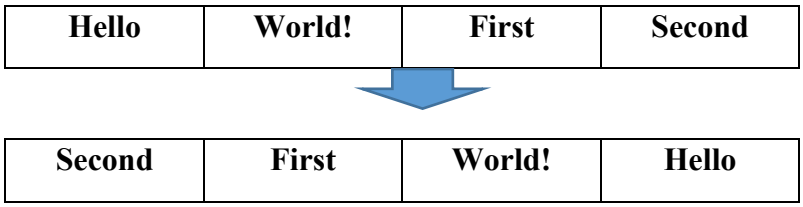
```
public int returnLargest(AVL T){

}
```

Part b. [2 points] The following AVL tree is given. Remove (1) and then (12). Show all rotations that apply.



Part c. [3 points] The undo function in a word-processor (example MS-Word) use a stack to reverse all words/tokens in a text-file. Write the undo method using java, this method takes an array of String tokens as a parameter. The method returns the array with items in the reverse order. You must demonstrate the use of a Stack. Example




```
public String [] unDo(String [] Tokens){

}
```

Part d. [3 points] Naif is the student chair of the IEEE Branch at Prince Sultan University. He stores the information for all student members in a doubly linked list D. In each node, he stores, 1. Student Name, 2. Student ID, 3. Phone Number, in addition to the next and prev pointers. He has a list of 100+ students. If a student's IEEE membership expires, Naif would remove the student at position K. Write a method called removeIEEE that removes the node at position K from the list D.

```
public boolean remove(DList D, int K){

}
```

Question 3. [5 + 5 = 10 points]

Part a. [5 points] Give the run-time (Big Oh notation) for the following tasks

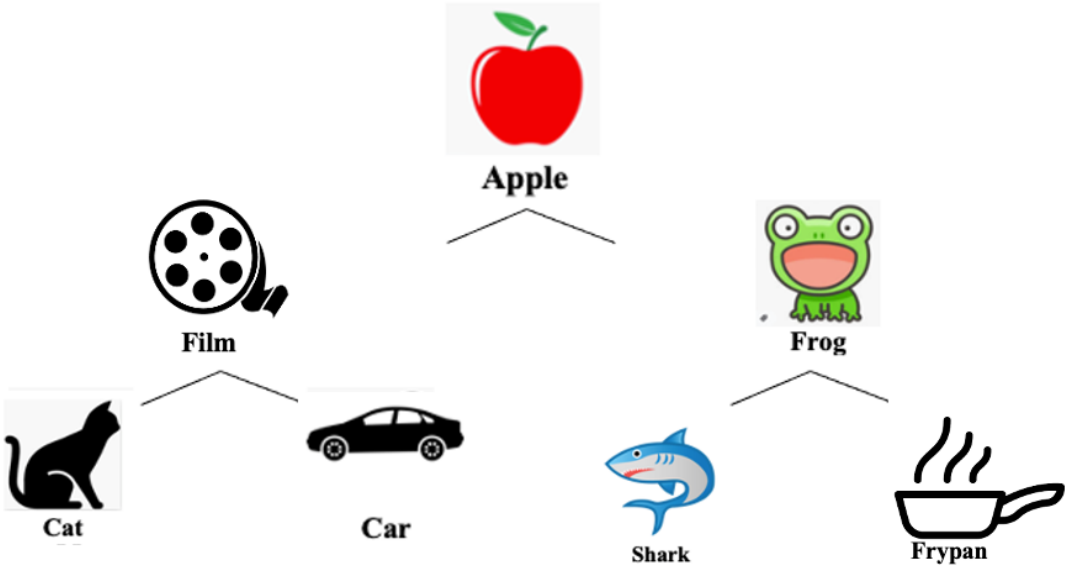
Number of comparisons done soring the data {B, C, G, H, K, L, M, Z} using Selection sort.	
Number of comparisons done soring the data {B, C, G, H, K, L, M, Z} using Insertion sort.	
Number of comparisons done searching for X in the data {B, C, G, H, K, L, M, Z} using Binary Search algorithm.	
Number of comparisons done Searching the value in the root of a AVL tree	
Emptying the max-heap by removing all elements from it.	
Searching for an element that is not available in a AVL tree.	
Amount of space needed to complete quick sort on a single dimensional array of size n.	
Searching for an element in a Hash table with uniformly distributed data	
Assuming that there are *n* number of vertices in a Graph; the size of adjacency matrix	
Runtime for sorting a pre-sorted array using the quick-sort algorithm	

Part b. [5 points] Sort the elements of the following array using quick-sort approach. Show all operations (pivot positions etc).

Cat	Bit	Hat	Mat	Rat	Bat	Hot	Bot	Sit

Question 4. [4 + 4 + 2 = 10 points]

Part a. [4 points] A min-heap of labels (key) and images (value) is given below. Each image is annotated with a String text-label which is to be used to compare items (String compare).



Remove an image from this heap. What image is returned? Show the resulting heap. (No need to draw pictures, only show the keys in the heap)

Add  **(Bat) to this heap. Show the resulting heap.**

Part b. [4 points] Find the Asymptotic complexity (Big O) for the following algorithms by computing the number of primitive operations:

`int k = 0;`
`for (i = n; i >= 0; i--) {`
`  for (j = 0; j <= i; j++) {`
`  k = k + i + j;`
`  }`
`}`	

`int k = 0, m = 0;`
`for (i = 1; i <= n; i++) {`
`  if(i%2==0)`
`  k++;`
`}`
`for (j = 1; j <= n; j++) {`
`  if(j%2==1)`
`  m++;`
`  }`	

`public void Myst(int arr[], int F, int L, int K){`
`  int m = (F + L)/2;`
`  while(F <= L ){`
`  if (arr[m] < K){`
`  F = mid + 1;`
`  }`
`  else if (arr[m] == K){`
`  System.out.println("found at:" + mid);`
`  break;`
`  }else`
`  {`
`  L = m - 1;`
`  }`
`  m = (F + L)/2;`
`  }`
`  if (F > L){`
`  System.out.println("not found!");`
`  }`
`}`	

Part c. [2 points] Give the run-time Asymptotic complexity (Big O) for the following recursive algorithm. Show the trace (recursion tree) for the call `fun(4)` .

```
public int fun(int n)
{
    if (n <= 1)
       return n;
    return fun(n-1) + fun(n-2);
}
```

Note: In case you forgot your ABCs! use this to help you determine the order

-End of Exam

<Scratch sheet. DO NOT detach>