Name: Hoang D. **ID#** 015218888

COMPUTING II

MIDTERM 1

You have 50 minutes to complete the midterm. The total number of points is 88. You are not allowed to use any books, notes, calculator, or electronic devices. Write your answer carefully and clearly. Incorrect answers will receive little to no points. When you are asked to write a program the program should be clear and include indentations and comments to make it easier to read. Be sure to ***state any assumptions*** on which you have based your answers.

Problem Number(s)	Possible Points	Earned Points
1	14	14
2	14	12
3	10	8
4	5	4
5	10	7
6(a)	10	9
6(b)	6	6
6(c)	17	1
7	2	2
8(*Extra*)	2	1
	TOTAL POINTS 88 (+2)	64

14

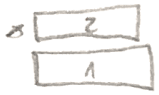
Stack

1. **(14 points)** Draw a sequence of diagrams, one for each problem segment, that represent the current state of the stack after each labeled set of operations. If an operation or instruction produces output then indicate what that output is. **hStack** is the handle of a stack opaque object that can hold characters.

```
hStack = stack_init_default();

a. stack_push(hStack, '1');
b. stack_push(hStack, '2');
c. printf("%d ", stack_top(hStack)); stack_pop(hStack);
d. printf("%d ", stack_top(hStack)); stack_push(hStack, '3');
e. stack_push(hStack, '4');  stack_push(hStack, '5');
f. printf("%d ", stack_top(hStack)); stack_push(hStack, '6');
g. stack_pop(hStack); stack_push(hStack, '5');

stack_destroy(&hStack);
```

a)

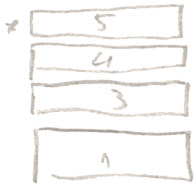
b)



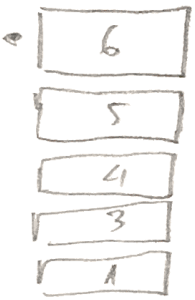
c) print the top items of the stack : 2
and then remove it

d) print the top item of stack : 1 (1)
push item 3 to stack

e)



f) print the top item of stack : 5 (1)
push item 6 to stack



g)

Queues

12

2. (14 points)

Draw a sequence of diagrams, one for each problem segment, that represent the current state of the queue after each labeled set of operations. If an operation or instruction produces output then indicate what that output is. We will use the function enqueue to add to the queue and serve to remove from the queue. **hQueue** is the handle of a queue opaque object that can hold characters.

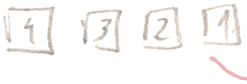
```
hQueue = queue_init_default();

a. queue_enqueue(hQueue, '1');
b. queue_enqueue(hQueue, '2');
c. printf("%c ", queue_front(hQueue)); queue_enqueue(hQueue, '3');
d. printf("%c ", queue_front(hQueue)); queue_enqueue(hQueue, '4');
e. queue_dequeue (hQueue); queue_dequeue (hQueue);
f. printf("%c ", queue_front(hQueue)); queue_enqueue(hQueue, '5');
g. queue_dequeue (hQueue); queue_dequeue(hQueue);

hQueue->destroy(&hQueue);
```

a) 

b) 

c) print the front Item in queue: 2 (1)

d) print ~~the~~ front Item in queue 3 (1)



e) 

f) print the front Item in queue 4. (3)

g) 

8

Expression Evaluation

3. **(10 points)** Evaluate the following expressions assuming 32 bit integers and 32 bit pointers. Variables are declared as listed but after some unknown number of operations the current state of the memory is given by the supplied memory diagram.

```
struct node
{
        int data;
        struct node* other;
};
typedef struct node Node;
Node v;
Node* p;
```

Variable Name /	Address	Memory Value
v	8000	3
	8004	9000
	8008	9004
p	8012	9028
	8016	9032
	8020	9020
	...	...
	9000	42
	9004	9016
	9008	5
	9012	100
	9016	87
	9020	9008
	9024	101
	9028	1
	9032	8000
	9036	9016

a. `v.data;` 3 ✓

b. `v.other->data;` 42 ✓

c. `p->other->data;` 3. ✓

d. `p->other[2]->data;` ~~87~~ invalid.

e. `p->other->other->other->data` 87 ✓

Stack

4. **(5 points)** Write a C declaration for a stack of floating point numbers stored in an **array**.

4

```
struct stack
{ int capacity;
  int size;
  float data;
}

typedef struct stack Stack;
```

⇒ answer

-1

5. **(10 points)** Using the above stack declaration, write a C function that implements the pop operation including printing a message and exiting if stack underflow occurs. You may assume the existence of an **`empty()`** function that takes a pointer to a stack as an argument.

⇒ answer

```
Node pop (Stack[] p'Head) sizeof)
{ ~~Node *temp = (Node *) p'Head;~~
  if (empty(5))
  { printf ("Nothing to pop from the stack \n");
    return FAILURE;
  }
  else
  { ~~temp = p'Head->next;~~ for(int i=0; i<n; i++)
    free (p'Head); {Head[i] = Head[i]}
    ~~p'Head = temp;~~ size--;
}
```

-3

④

```
typedef struct
{
  float array[10];
  int top;
}stack;
stack* myStack = malloc(sizeof(stack));
myStack->top = -1;
```

⑤

④

```
void pop(stack* myStack)
{
  if(empty(myStack) == 1)
    printf("ERROR: Stack underflow.\n");
  else
  {
    printf("The popped element is: %f \n",
                myStack->array[myStack->top]);
    myStack->top--;
  }
}
```

Linked list

6. **(33 points)** Given the following:

```
typedef node Node;
struct node {
  int data;
  Node* next;
};
```

a. **(10 points)** Write a function called destroy that takes a Node pointer to the head of a list and will free up the memory associated with each node in the entire list.

```
void destroy (Node ** pHead)
{ Node * temp = (Node*) *pHead;

  while (temp != NULL)
  { temp = pHead -> next;
    free (pHead);
    pHead = temp;
  }
    return -1
}
```

b. **(6 points)** Write a recursive function called sum that given a Node pointer to the head of a list will return the sum of all data in the linked list.

```
int sum ( Node * pHead)
{ Node * temp = Node *( *pHead );
  if ((temp -> next->data)==NULL)
  { return temp -> data;
  }
  return (temp -> data) + sum ( temp-> next);
}
```

1/7

c. **(17 points)** Write a function called **`copy_list`** that, given a Node pointer to the head of a list will return a Node pointer containing the address of the head node of a new list that is an exact copy of the original list. Your copy should be independent of the first list and not share any nodes. You may write an iterative or recursive version of your function.

```
Node * copy_list (Node ** pHead).
{ Node * temp = (Node)* pHead;
  Node* newNode = (Node*) malloc (sizeof (Node));
  While ((temp -> next -> data) != NULL)
  { newNode = temp;
    newNode = newNode.next }.
return newNode;
```

7. **(2 points)** Following is an ***incorrect*** pseudocode for the algorithm which is supposed to determine whether a sequence of parentheses is balanced:

```
declare a character stack
while (more input is available)
{
   read a character
   if (the character is a '(' )
      push it on the stack
   else if (the character is a ')' and the stack is not empty)
      pop a character off the stack
   else
      print "unbalanced" and exit
}
print "balanced"
```

Which of these unbalanced sequences does the above code think is ***balanced***?

a. (((()
b. ())(()
c. (()))
d. (()))()

Extra

8. **(2 points)** What is a stack?

stack is a data structure that we to store datas in a particular order two function that are used to insert and remove items from stack. It follow the principle First in Last out

Pop and push -1

Name: Hoang Do ID# 015218884

COMPUTING II

MIDTERM 2

You have 50 minutes to complete the midterm. The total number of points is 80. You are not allowed to use any books, notes, calculator, or electronic devices. Write your answer carefully and clearly. Incorrect answers will receive little to no points. When you are asked to write a program the program should be clear and include indentations and comments to make it easier to read. ***Be sure to state any assumptions on which you have based your answers.***

Problem Number(s)	Possible Points	Earned Points
1	20	20
2	12	12
3	15	10
4	5	5
5	10	5
6	15	12
7	3	3
	TOTAL POINTS 80	67

```
for (j = 1 ; j < n-1 ;
    smallest-1 ← j
    for (i = j+1 to n
        i)
```

1. (20 points) **Sorting Algorithms**: Define the following sorting algorithms using text and or pseudo code.
   a. Selection Sort

- start from the first element of array, find the small element in the array and swap them
- move to the next element and repeat the code

```
int i, min;
for(i=0; i<size; i++){
    min = find_min_index(a, size, i);
    swap(&a[i], &a[min]);
}
```

+5

   b. (5 points) Bubble Sort

- compare a pair of nodes starting from the first node
  + if the first node in the pair greater than the second node, then swap
  + else move on next element in the array
- repeat again

```
int i, j;
for(j=0; j<size; j++){
    for(i=0; i<size; i++){
        if(a[i] > a[i+1]){
            swap(&a[i], &a[i+1]);
        }
    }
}
```

+5

c. (5 points) Insertion Sort

```
int i, j;
for (i = 0; i < size; i++) {
    j = i;
    while (j - 1 >= 0 && a[j] < a[j-1])
    {swap (&a[j], &a[j-1]);
     j--;
    }
}
```

+5

- start from first elem of array, compare to the next element, swap if the second element less than first one.
- compare them to the next elem and repeat until finish the array

d. (5 points) Shell Sort

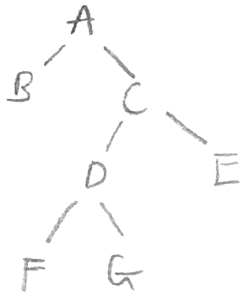
```
int i, j, h
h = size/3 + 1;
do{
  h--;
  for(i = h; i < size; i++){
     j = i;
     while( j - h >= 0 && a[j] < a[j-1]){
        swap ( &a[j], &a[j-1]);
        j--;
     }
  }
} while ( h != 1);
```

+5

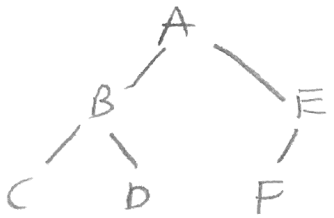
2. (12 points) **Full vs Complete Binary Trees**

12

a. (3 points) Give an example of a binary tree that is full but not complete.

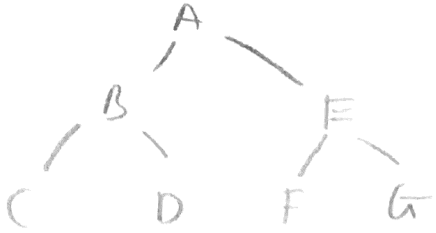


b. (3 points) Give an example of a binary tree that is complete but not full.



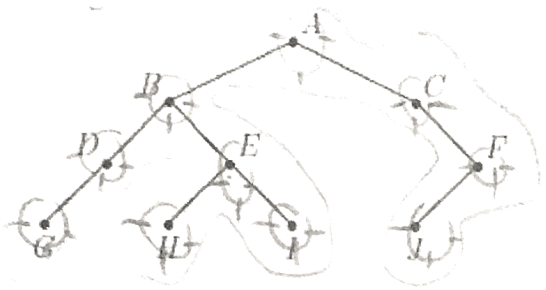
c. (3 points) Give an example of a binary tree that is neither full nor complete.

d. (3 points) Give an example of a binary tree that is both full and complete.



3. (15 points) List the sequence of nodes visited by pre-order, in-order, and post-order traversals of the following tree:

10



(a) (5 points) Give the output for a **preorder** traversal calling visit.

-2 A B D G H I E J C F J.

E H I

(b) (5 points) Give the output for an **inorder** traversal calling visit.

G D B H I E J F C A

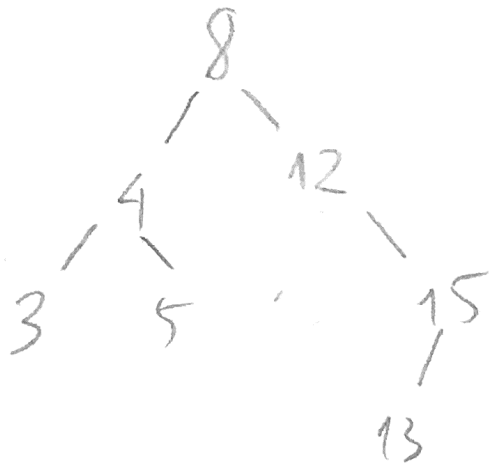
-3 E I A C J F

(c) (5 points) Give the output for a **postorder** traversal calling visit

G D H I E B J F C A.

4. (5 points) Assuming the following values arrive in the order they appear and are inserted into a ***binary search tree***, show the resulting tree.

   8, 12, 4, 15, 3, 5, 13



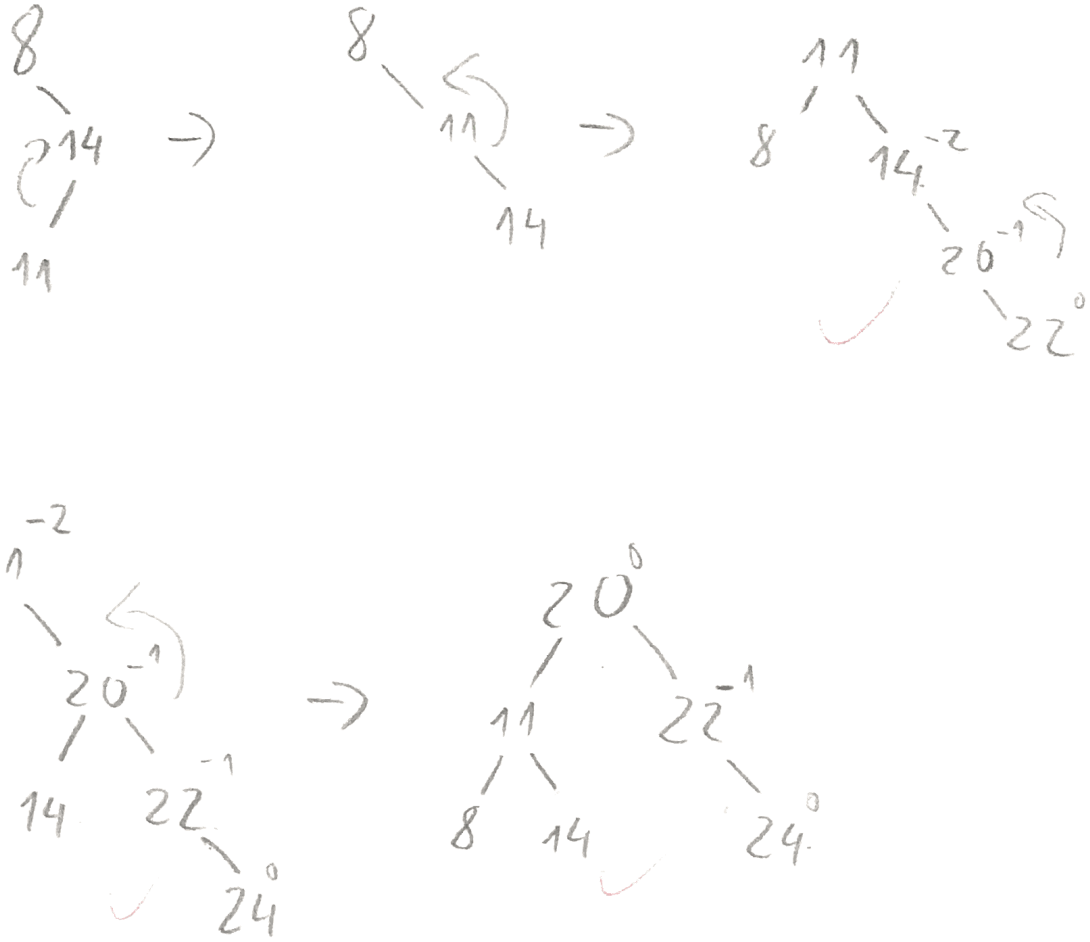
5. (10 points) Write down a ***recursive function to destroy a tree***. The function takes a node pointer. The recursive function declaration is:
`void binary_tree_destroy(BinaryNode pNode);`

```
void binary_tree_destroy (BinaryNode *pNode) {
    if (pNode != NULL) {
        binary_tree_destroy ( &(pNode) -> left);
        binary_tree_destroy ( &(pNode) -> right);
        free (pNode);
        pNode = NULL;
    }
}
```

Đáp án

```
{
if (pNode == NULL)
    return;
binary_tree_destroy (pNode -> left);
binary_tree_destroy (pNode -> right);
free(pNode);
pNode = NULL; }
}
```

6. (15 points) Assuming the following values arrive in the order they appear and are inserted into ***AVL tree***, show your work and the resulting tree.
8, 14, 11, 20, 22, 24, 23



7. (3 points) What would be the output of the following program:

```
int fun(int n)
{
  if (n == 4)
        return n;
  else return 2*fun(n+1);
}
int main()
{
  printf("%d ", fun(2));
  return 0;
}
```

16

1) n = 2 → 2 × fun(3)
n = 3 → 2 × fun(4) stop
= 2 × 4

⇒ 2 × 2 × 4 = 16

2 × fun(3) = 2 × fun(2 × fun(4))
= 2 × 2 × 4

Heap (max)

Name: Hoang Do ID# 015218XX

COMPUTING II

Quiz 1

You have 15 minutes to complete the Quiz. The total number of points is 5. You are not allowed to use any books, notes, calculator, or electronic devices. Write your answer carefully and clearly. Incorrect answers will receive little to no points. When you are asked to write a program the program should be clear and include indentations and comments to make it easier to read. Be sure to state any assumptions on which you have based your answers.

Suppose you have a linked list that stores numbers 1, 2, 4, 5 :

head →1→2→4→5

Describe (draw/provide a pseudocode) step by step how you will insert a new node that holds number 3 into the list that the resulting list looks like this:

head →1→2→3→4→5

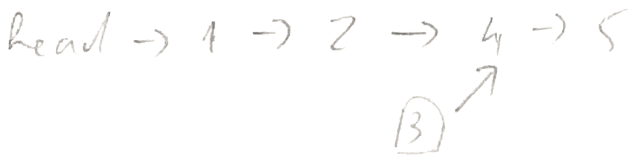
allocate memory
create a new node

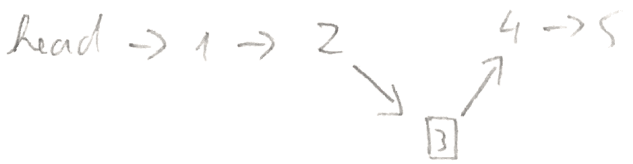
```
newNode -> data = 3;
currentNode = head;
while ((currentNode -> data) != 2)
    { currentNode = currentNode -> next; }
newNode -> next = currentNode -> next;
currentNode -> next = newNode;
```

(4,5)

head → 1 → 2 → 4 → 5

[3]





Name: Hoang Do ID# 015218888

COMPUTING II

Quiz 2

You have 15 minutes to complete the Quiz. The total number of points is 10. You are not allowed to use any books, notes, calculator, or electronic devices. Write your answer carefully and clearly. Incorrect answers will receive little to no points. Be sure to state any assumptions on which you have based your answers.

1. (10 points) **Expression Evaluation.** Evaluate the following expressions assuming 32 bit integers and 32 bit pointers. Variables are declared as listed but after some unknown number of operations the current state of the memory is given by the supplied memory diagram.

```
struct my_vector
{
      int size;
      int capacity;
      int* data;
};
typedef struct my_vector My_vector;
My_vector p;
My_vector* t;
```

Variable Name / Address	**Memory Value**
p 8000	3
8004	4
8008	9004
t 8012	9028
8016	10000
8020	9020
...	...
9000	42
9004	63
9008	5
9012	100
9016	87
9020	14
9024	101
9028	2
9032	3
9036	9016

a. `p.data;` 3 X 9004

b. `(p.data[2]) << 2;` 100 X 400

c. `&t;` 8012 ✓

d. `t->data[1];` 10000 X 14

e. `(*t).capacity;` 256 X 3

3/10

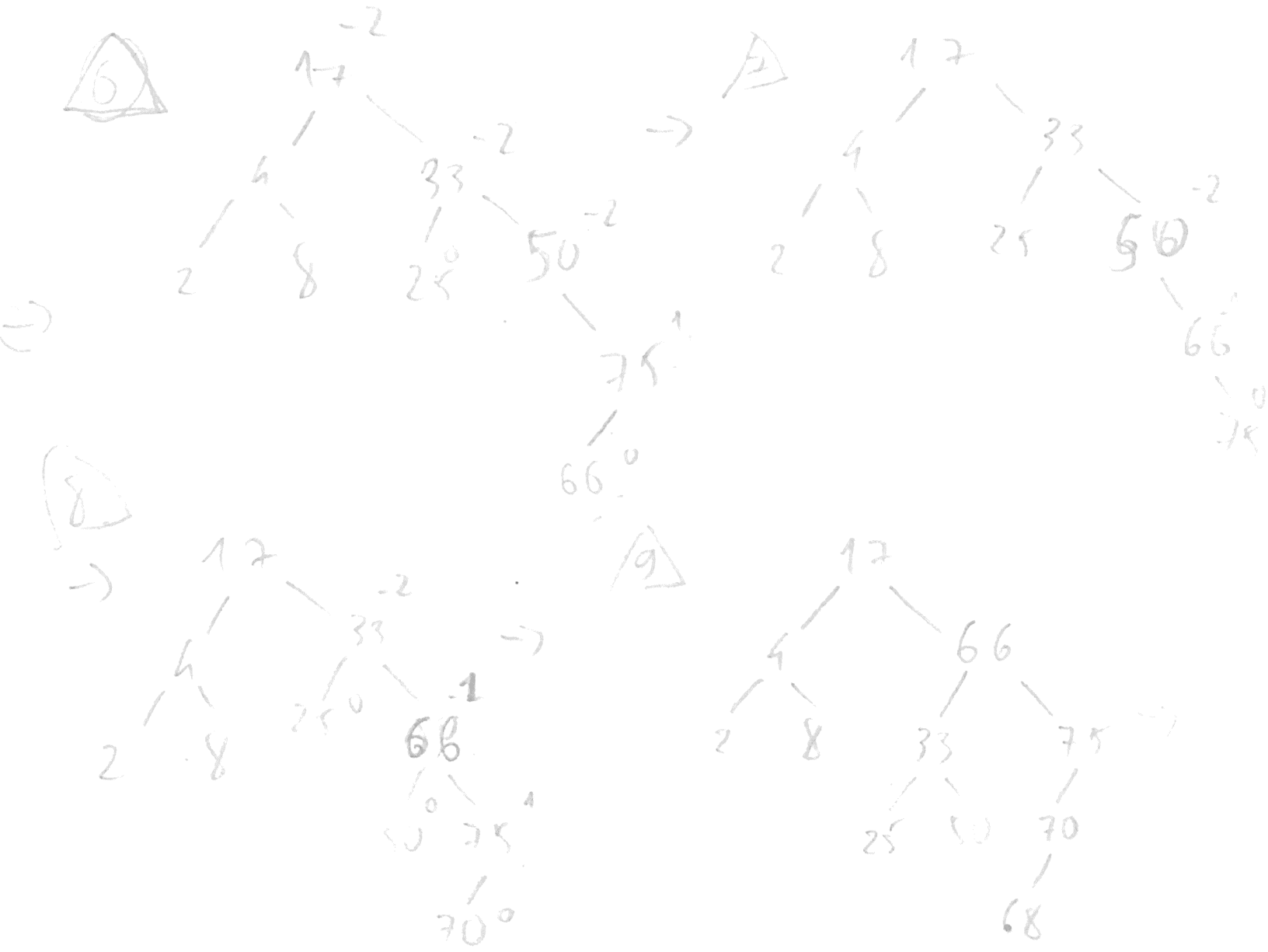
Name: Hoang Do ID# 015218888

COMPUTING II

Pop Quiz 3

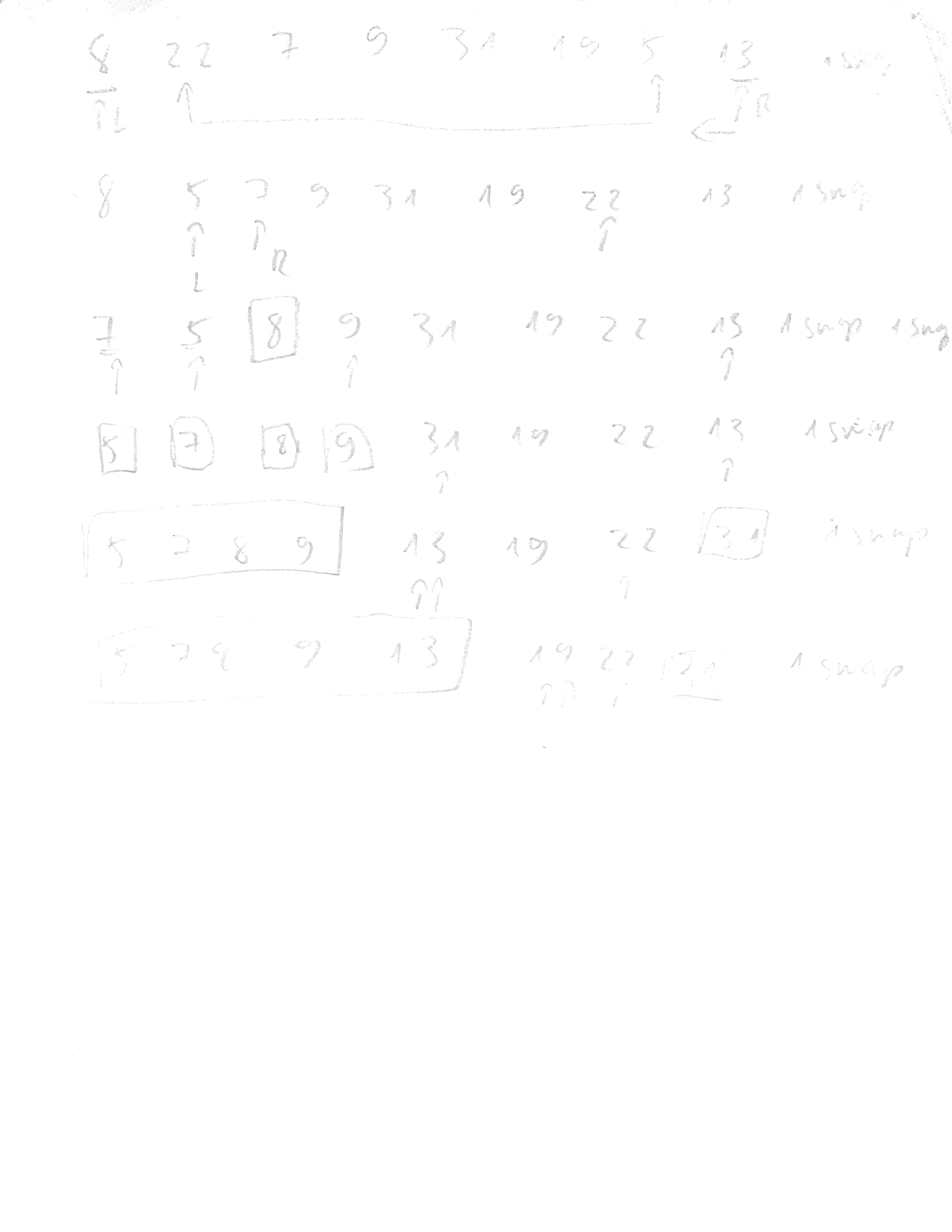
You have 15 minutes to complete the Quiz. The total number of points is 10. You are not allowed to use any books, notes, calculator, or electronic devices. Write your answer carefully and clearly. Incorrect answers will receive little to no points. When you are asked to write a program the program should be clear and include indentations and comments to make it easier to read. Be sure to state any assumptions on which you have based your answers.

3.5



10

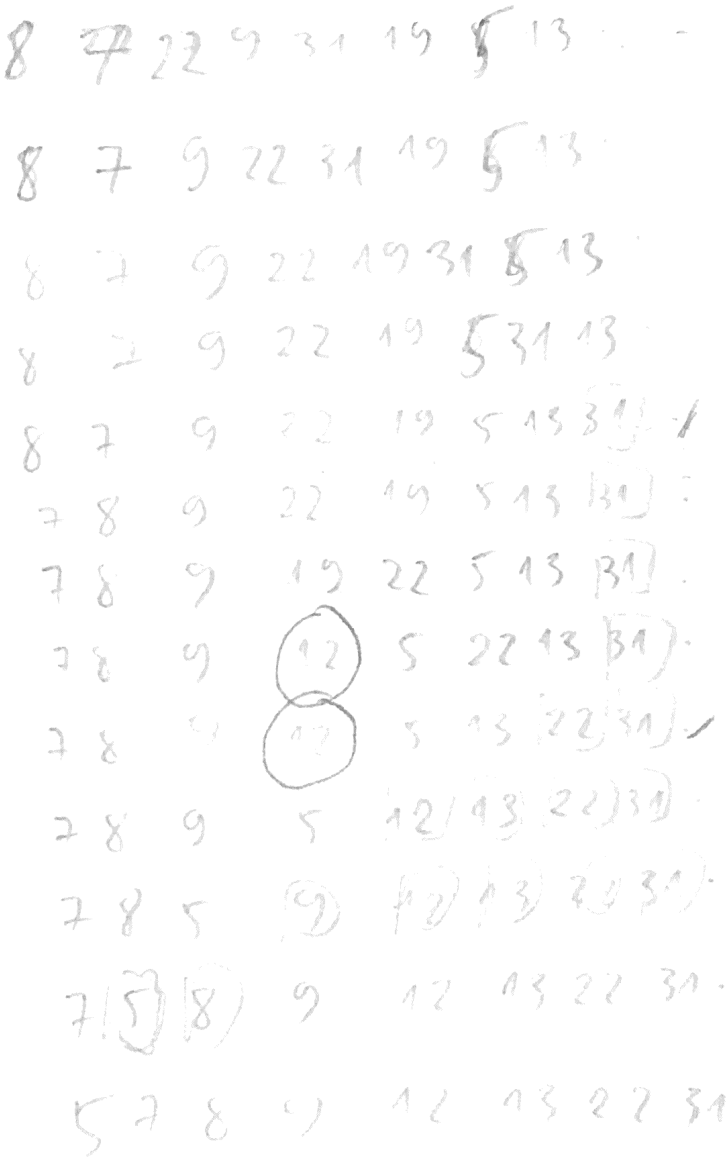
L - R



1. (2 points) How many numbers of swapping is needed to sort numbers

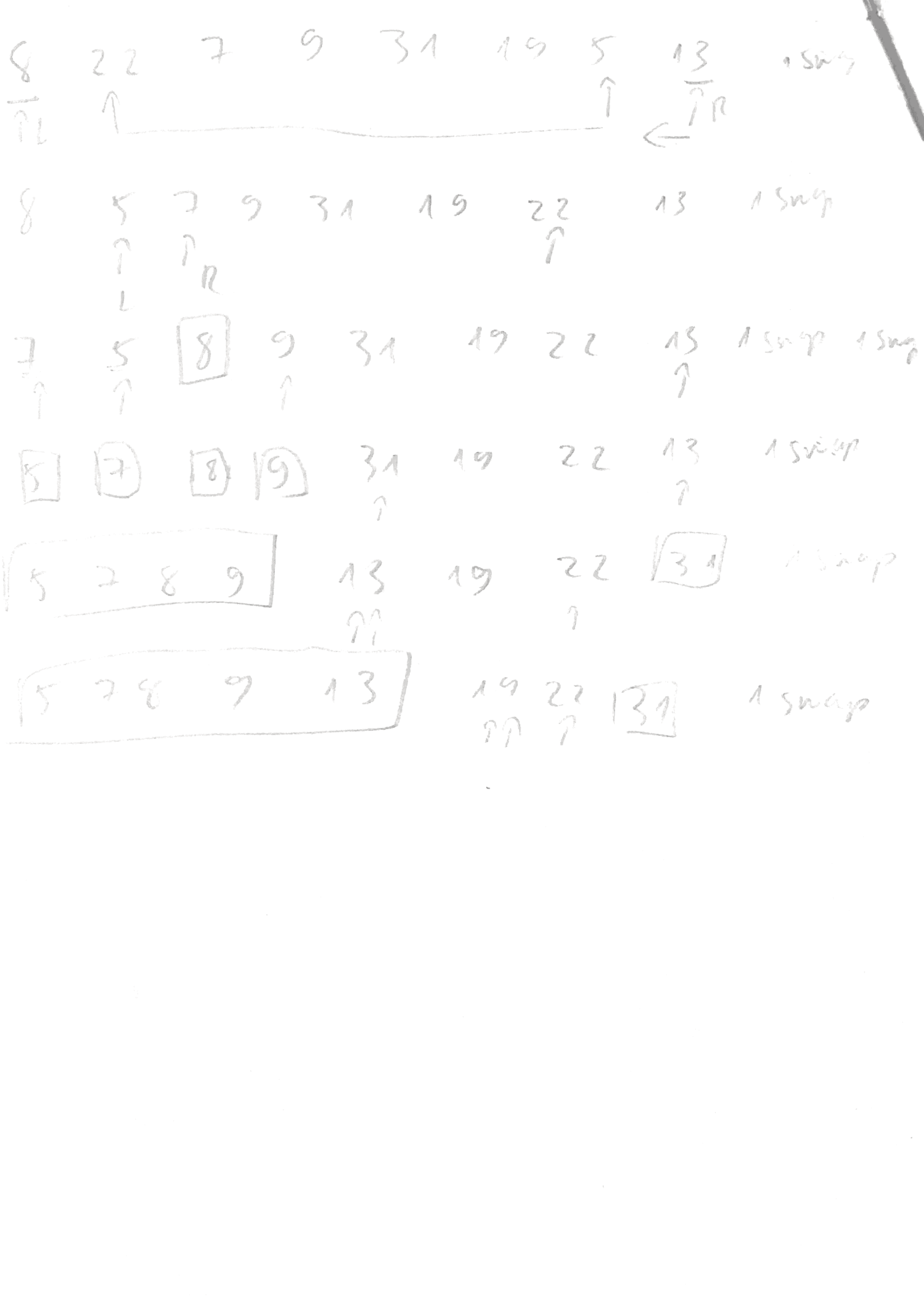
   8, 22, 7, 9, 31, 19, 5, 13

   in ascending order using bubble sort?



13 swaps is needed

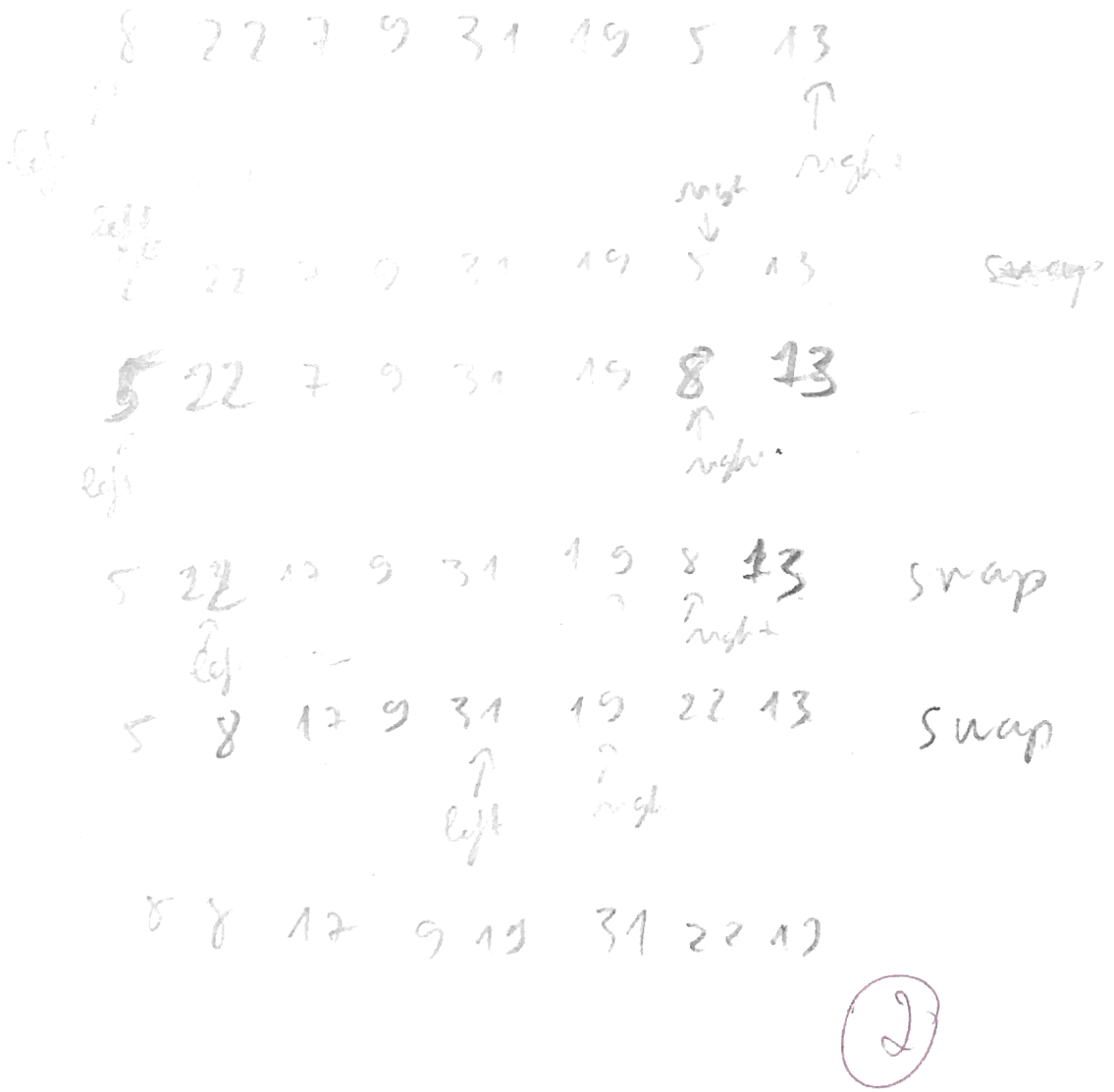
(1.5)



2. (8 points) Given the following list of elements, illustrate quicksort. Assume the pivot to be the first element in each list. I want to see the pivot, and each sub group/list clearly denoted. Follow the quicksort algorithm. Elements should be exchanged based on their value to the pivot and the correct placement (of the pivot) demonstrated.

   8, 22, 7, 9, 31, 19, 5, 13

   How many swaps are needed?



2

(1) LL Rotation

right rotation

(2) RR

left rotation

(3) LR

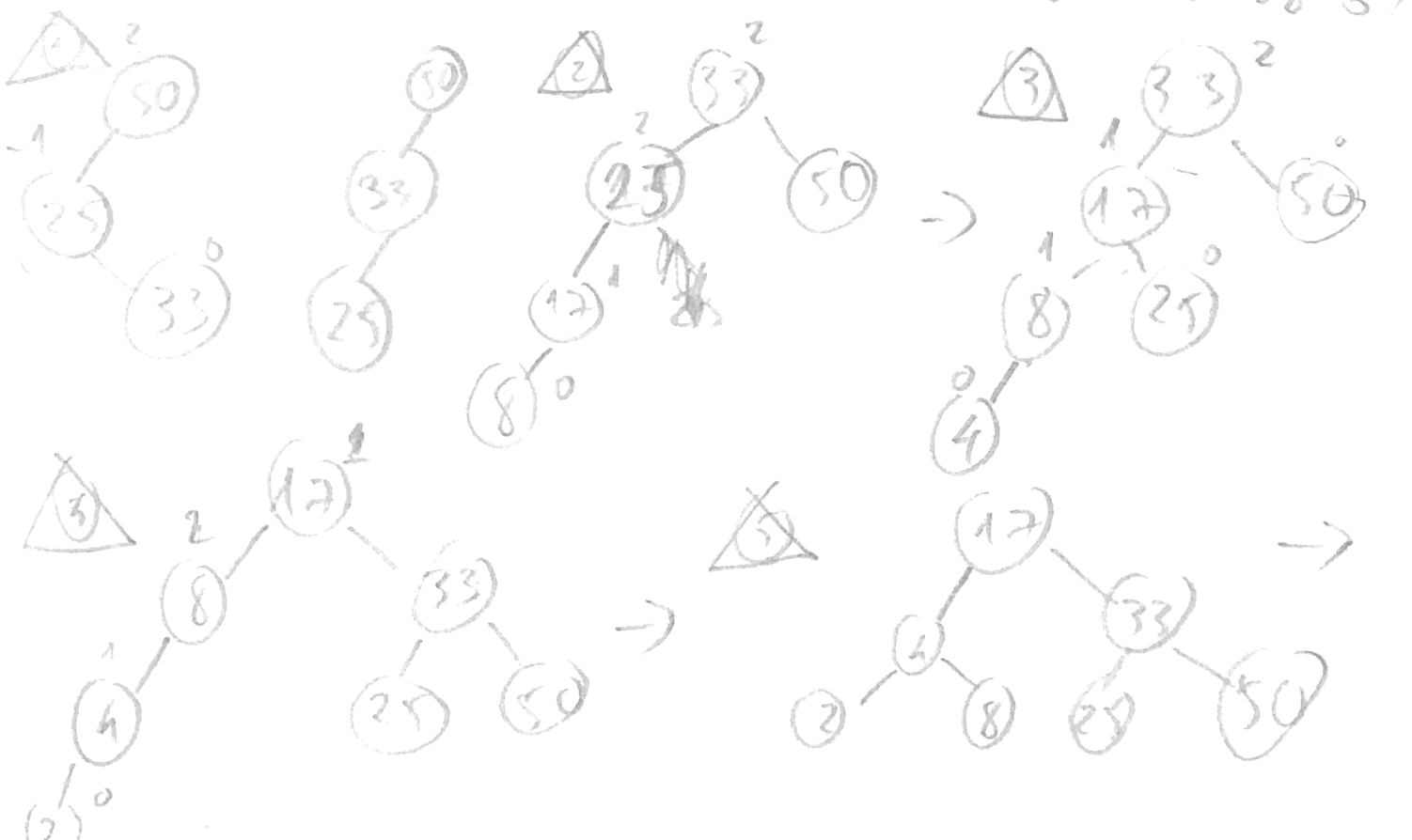
left a child
right a parent

(4) RL

right a child
left a parent

50
25 75
10 40 60

50 25 35 17 8 4 2 75 66 70 68 69



Name: Hoang Do ID# 01521888

COMPUTING II

Quiz 4

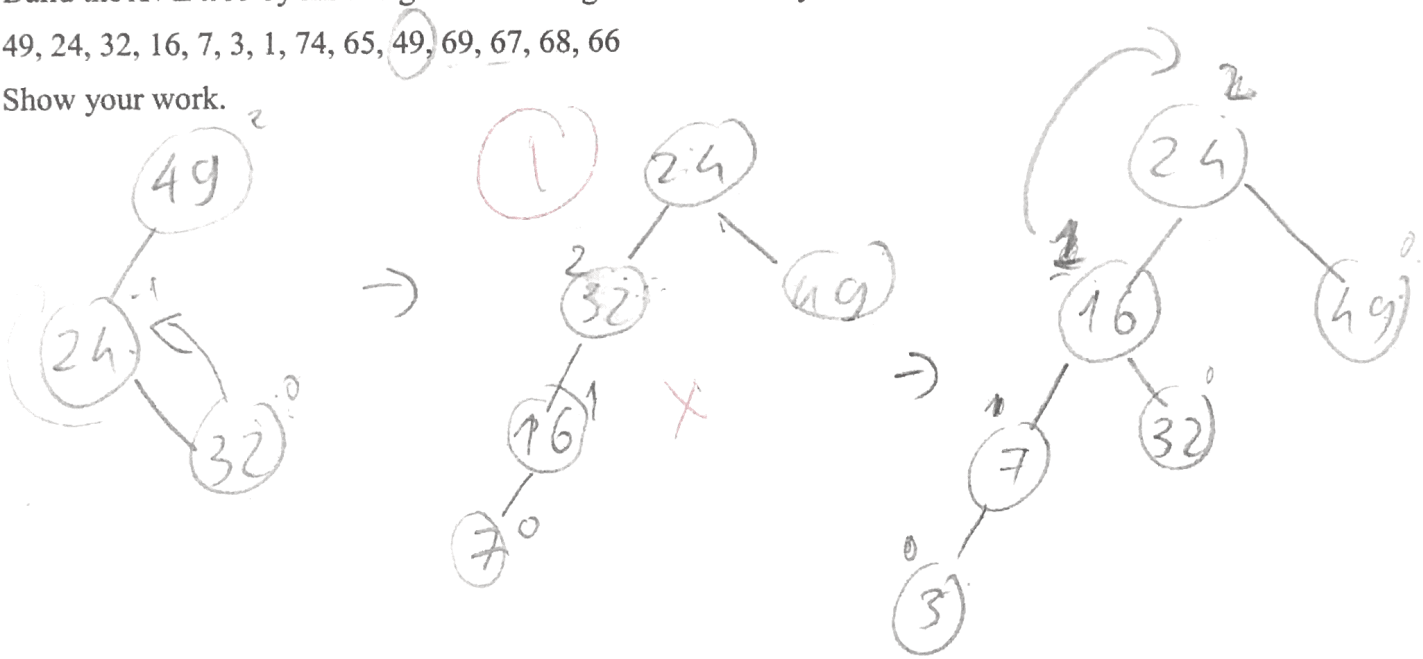
You have 20 minutes to complete the Quiz. The total number of points is 10. You are not allowed to use any books, notes, calculator, or electronic devices. Write your answer carefully and clearly. Incorrect answers will receive little to no points. When you are asked to write a program the program should be clear and include indentations and comments to make it easier to read. ***Be sure to state any assumptions on which you have based your answers.***

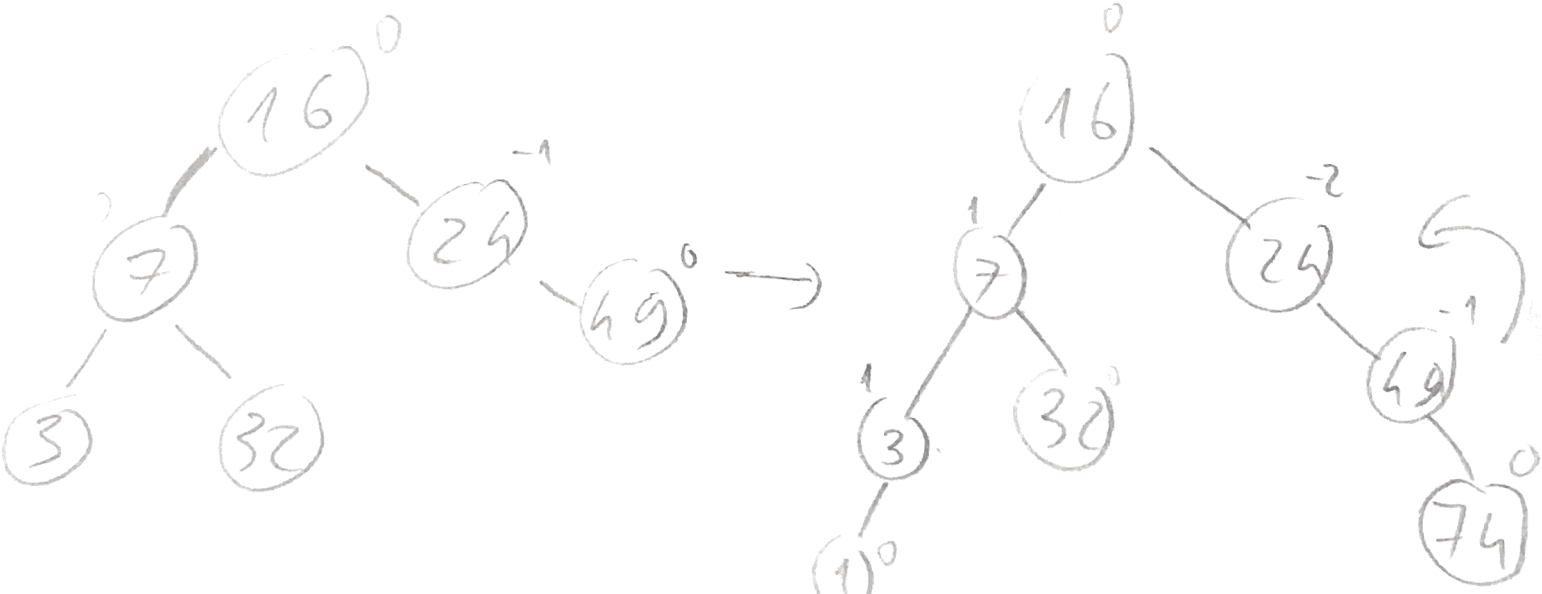
10

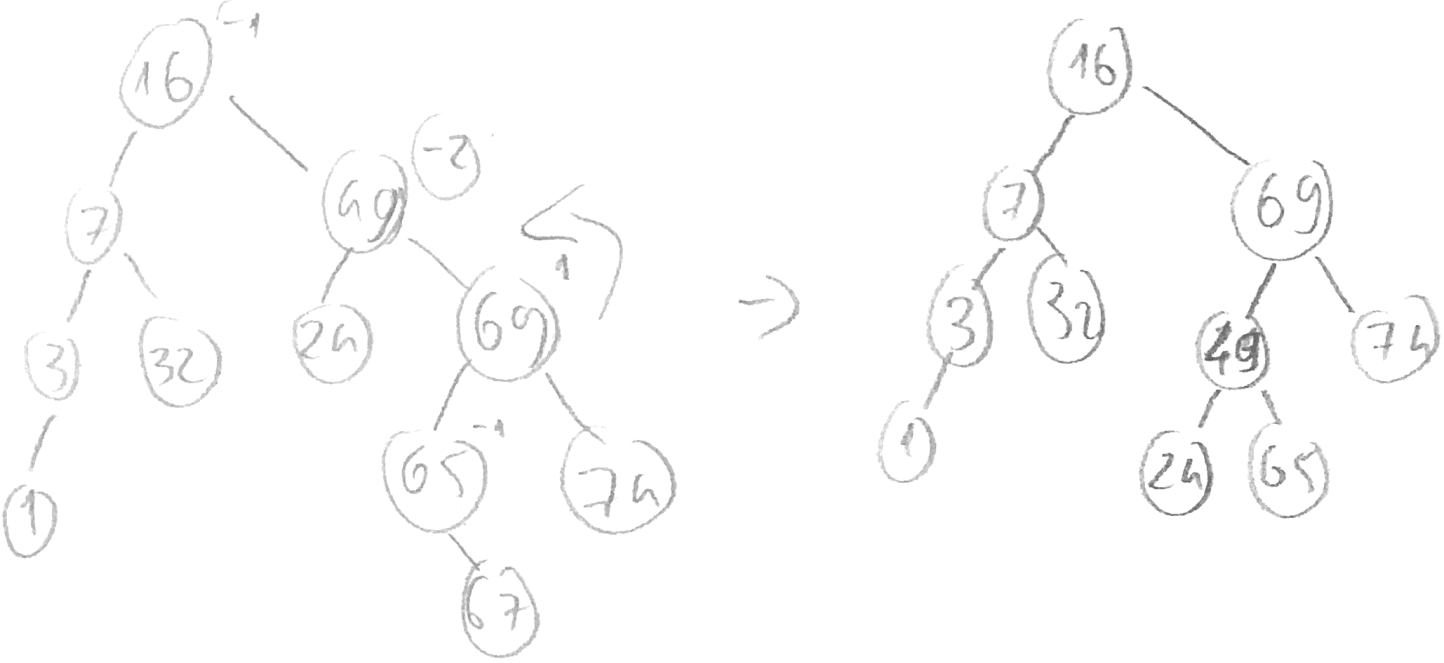
Build the ***AVL tree*** by inserting the following numbers one by one:

49, 24, 32, 16, 7, 3, 1, 74, 65, 49, 69, 67, 68, 66

Show your work.







Name: Hoang Do ID# 015218888

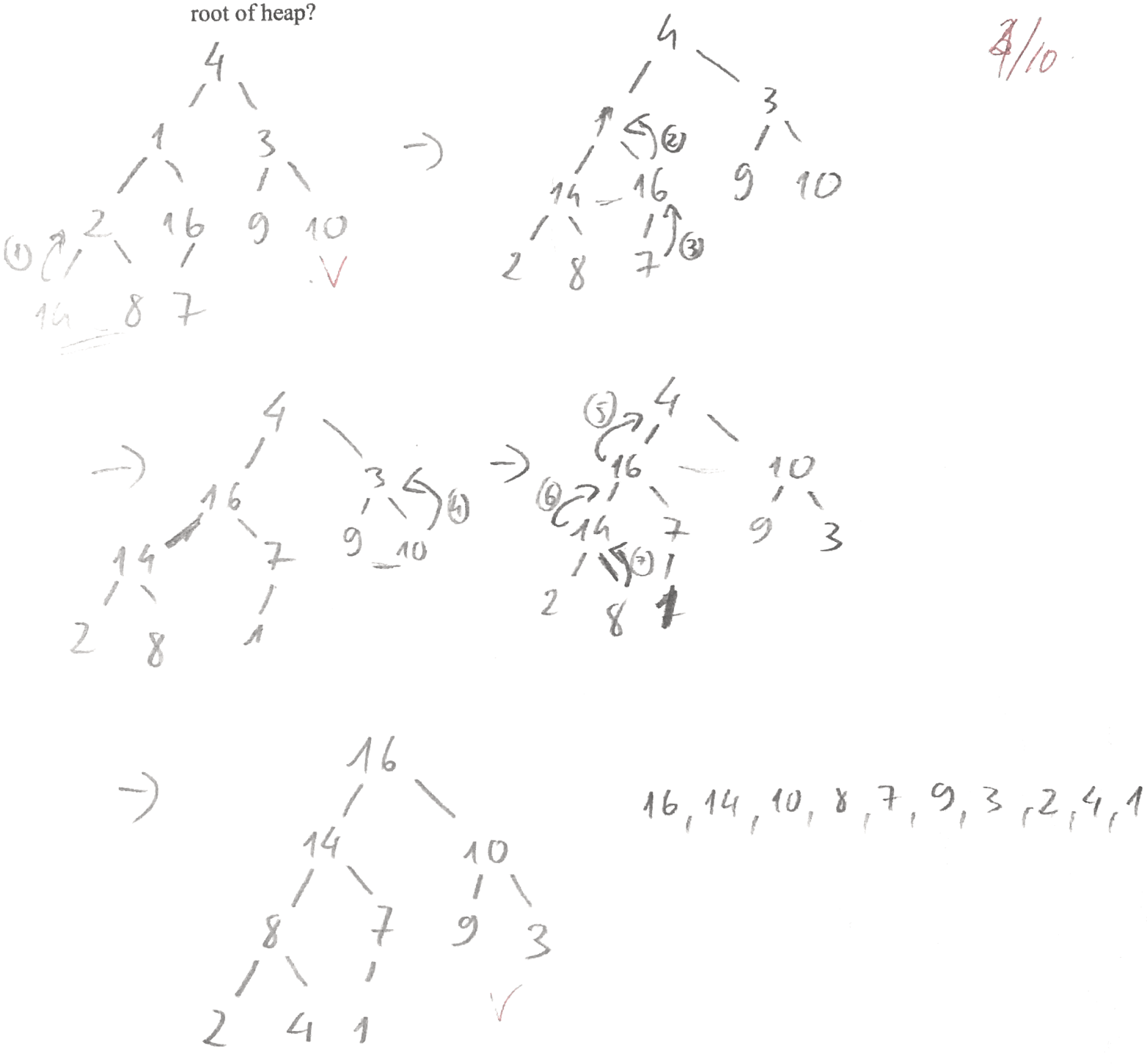
COMPUTING II

Pop Quiz 5

You have 15 minutes to complete the Quiz. The total number of points is 10. You are not allowed to use any books, notes, calculator, or electronic devices. Write your answer carefully and clearly. Incorrect answers will receive little to no points. Be sure to state any assumptions on which you have based your answers.

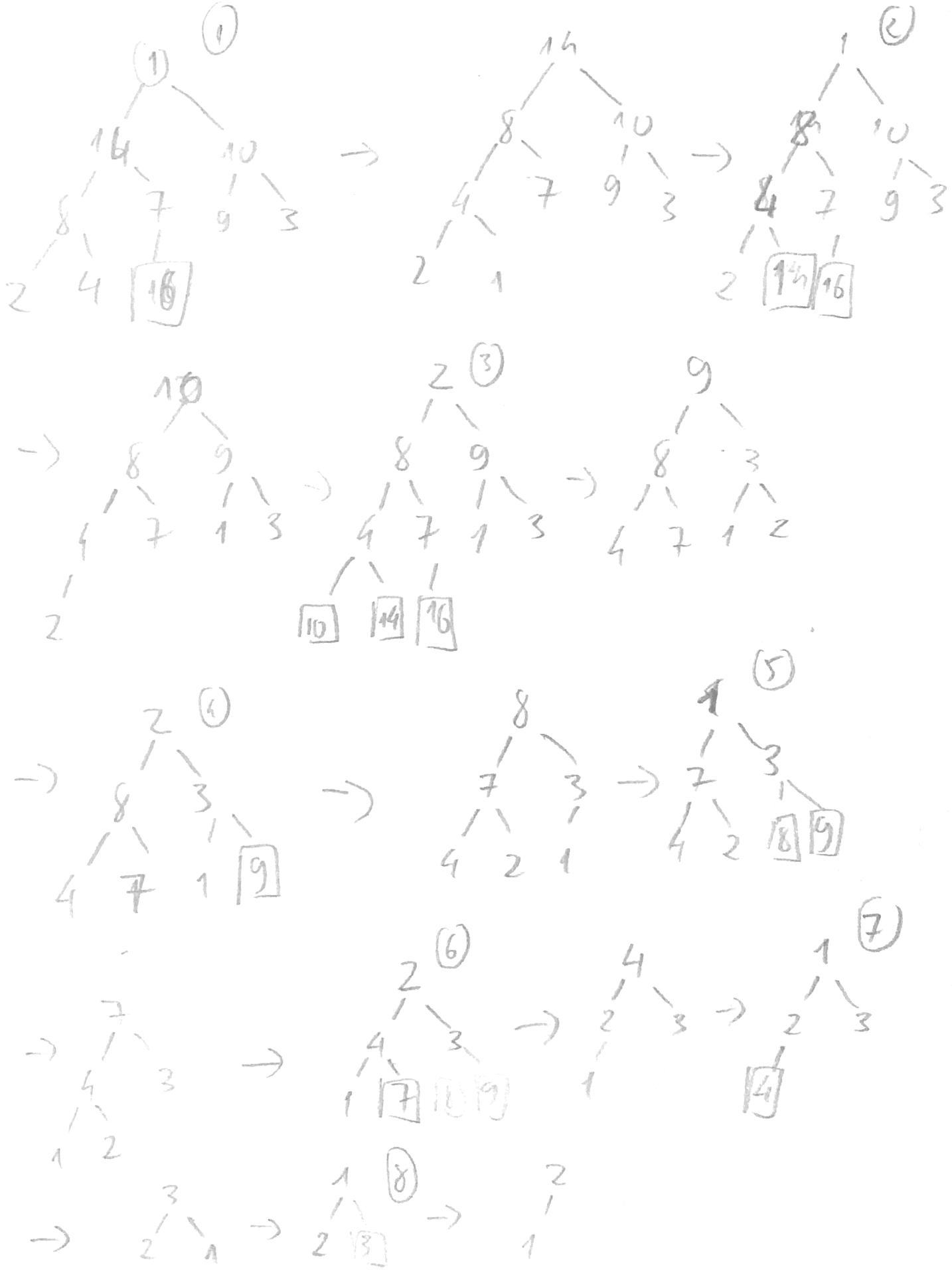
1. (10 points) **Heapsort.** Sort an array of integers
4, 1, 3, 2, 16, 9, 10, 14, 8, 7
using heapsort. Show the steps. How many heapify operation have been performed on root of heap?

~~8~~ 9/10



16, 14, 10, 8, 7, 9, 3, 2, 4, 1

⇒ 7 Heapify operations have been performed

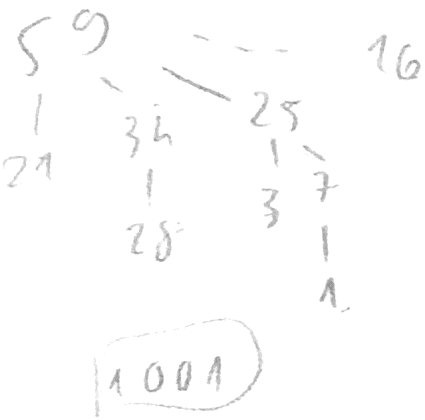

①
②
③
④
⑤
⑥
⑦
⑧

Name: Hoang Do ID# 015 21888

COMPUTING II

Quiz 6

You have 10 minutes to complete the Quiz. The total number of points is 5. You are not allowed to use any books, notes, calculator, or electronic devices. Write your answer carefully and clearly. Incorrect answers will receive little to no points. When you are asked to write a program the program should be clear and include indentations and comments to make it easier to read. Be sure to state any assumptions on which you have based your answers.



Computing II Quiz 6 Fall 2017

Insert next numbers into hash table: **2, 11, 3, 36, 5, 24, 46, 14** by using the next hushing functions

$h_1(x) = x\%(\text{HashTable\_size})$

$h_2(x) = 11 - (x\%11)$

and:

4/5

1. **(1 point) Chaining**

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]
11		2 → 24 → 46	3 → 36 → 14		5					

order ↓

2. **(1 point) Linear probing**

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]
11		2	3	36	5	24	46	14		

3. **(1 point) Quadratic probing**

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]
11	14	2	3	36	5	24	46			

$key + 1^2 \rightarrow key + 2^2 \rightarrow key + 3^2$

4. **(2 points) Double hashing**

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]
11		2	3	36	5	24	14	46		

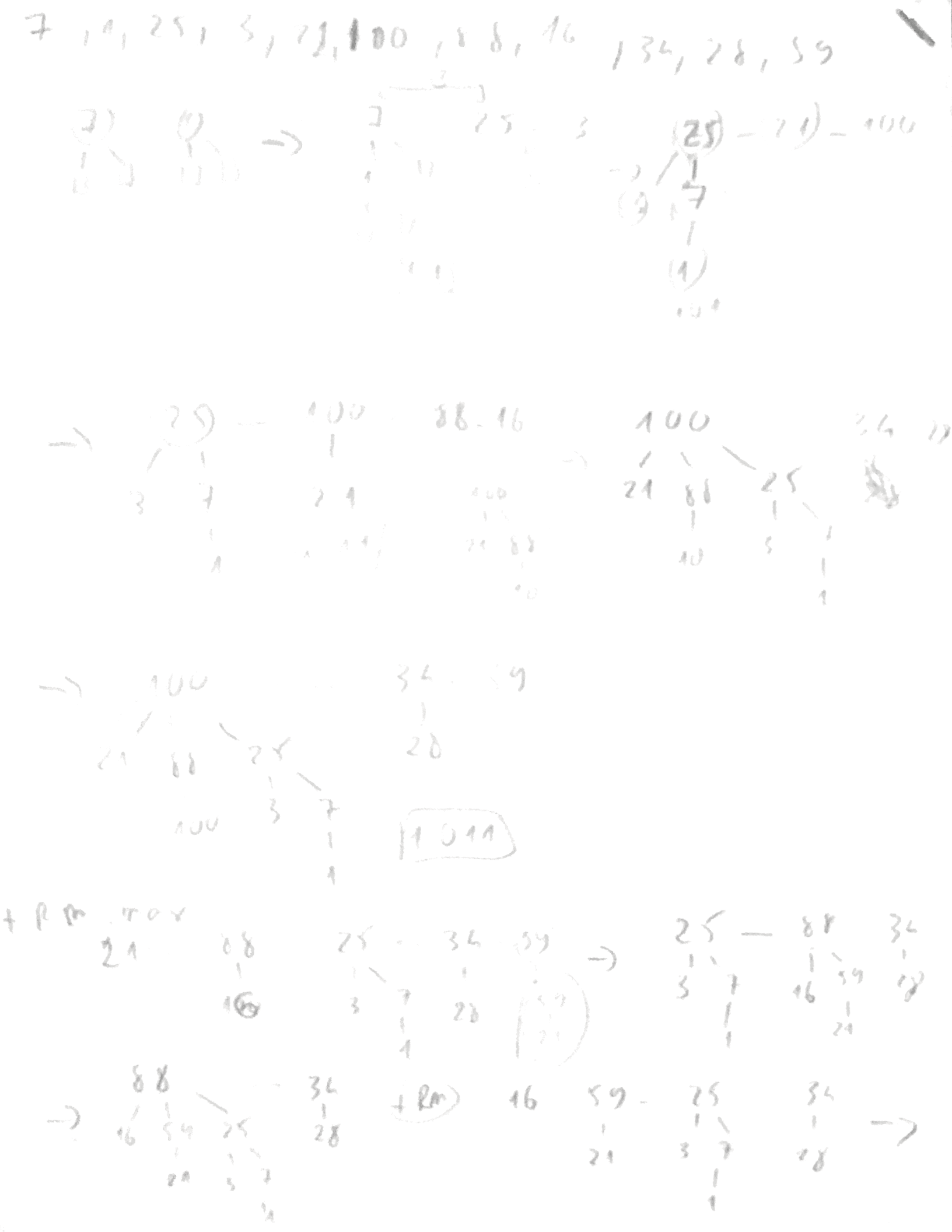
$key = h_1(x) + h_2(x)$

$key = h_1 + h_2$

$key = h_1 + 2h_2$

$+ 3h_2$

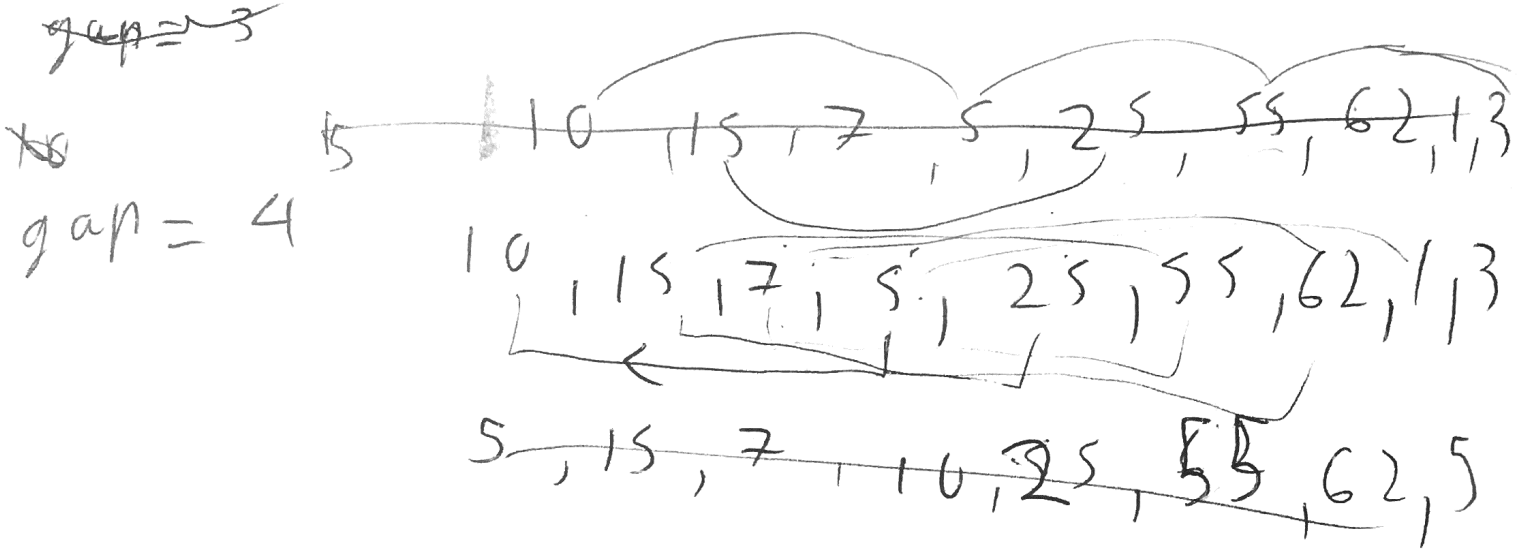
0	1	2	3	4	5	6	7	8	9	10
11		2	3		5		46	36	24	14



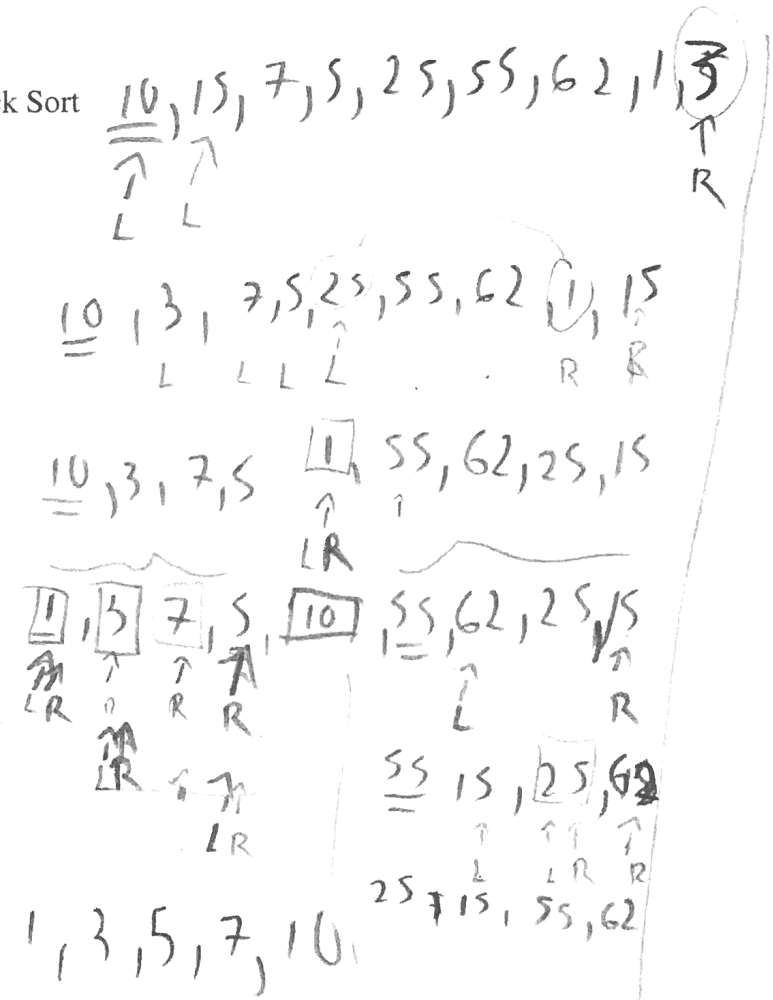
length 9

Given the following list of data: 10, 15, 7, 5, 25, 55, 62, 1, 3
Sort it by using

1. Shell Sort (show all steps)

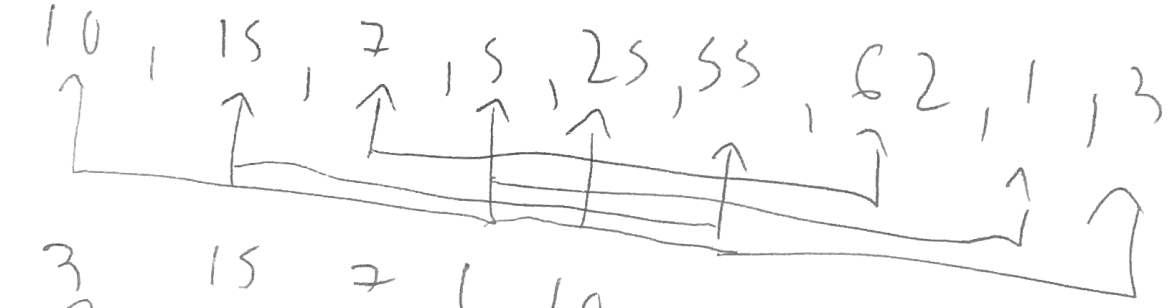


2. Quick Sort

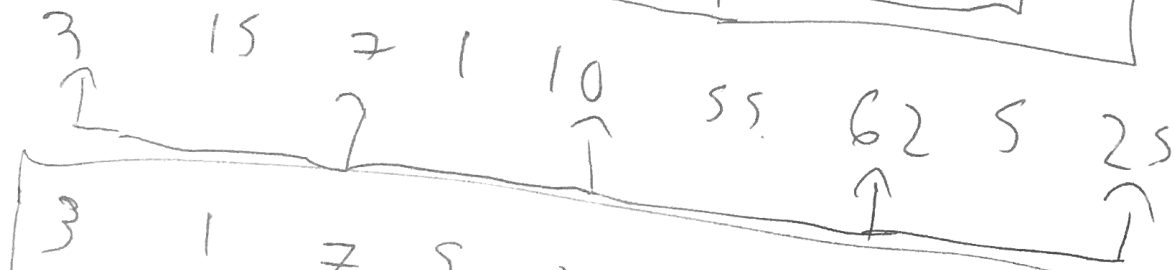


1/ Shell Sort:

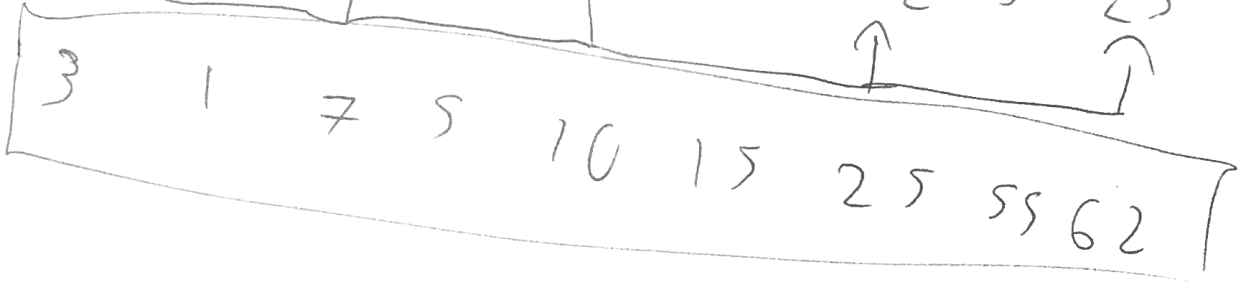
h = 4



1 step



h = 2



(2 step)

h = 1

2/ 10 , 15 , 7 , 5 , 25 , 55 , 62 , 1 , 3

Pivot ↑ L — R ↑

10 , 3 , 7 , 5 , 25 , 55 , 62 , 1 , 15

Pivot — L ↑ — R ↑

10 , 3 , 7 , 5 , 15 , 55 , 62 , 1 , 25

D ↑ — L ↑ — R ↑

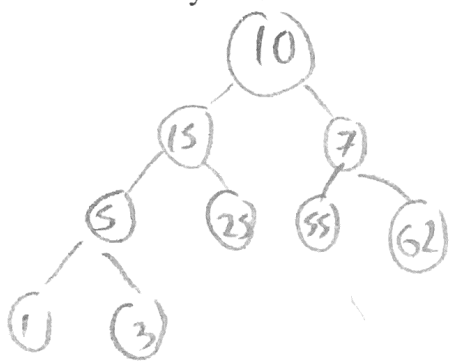
10 , 3 , 7 , 5 , 1 , 55 , 62 , 15 , 25

1 , 3 , 7 , 5 , 10 , 55 , 62 , 15 , 25

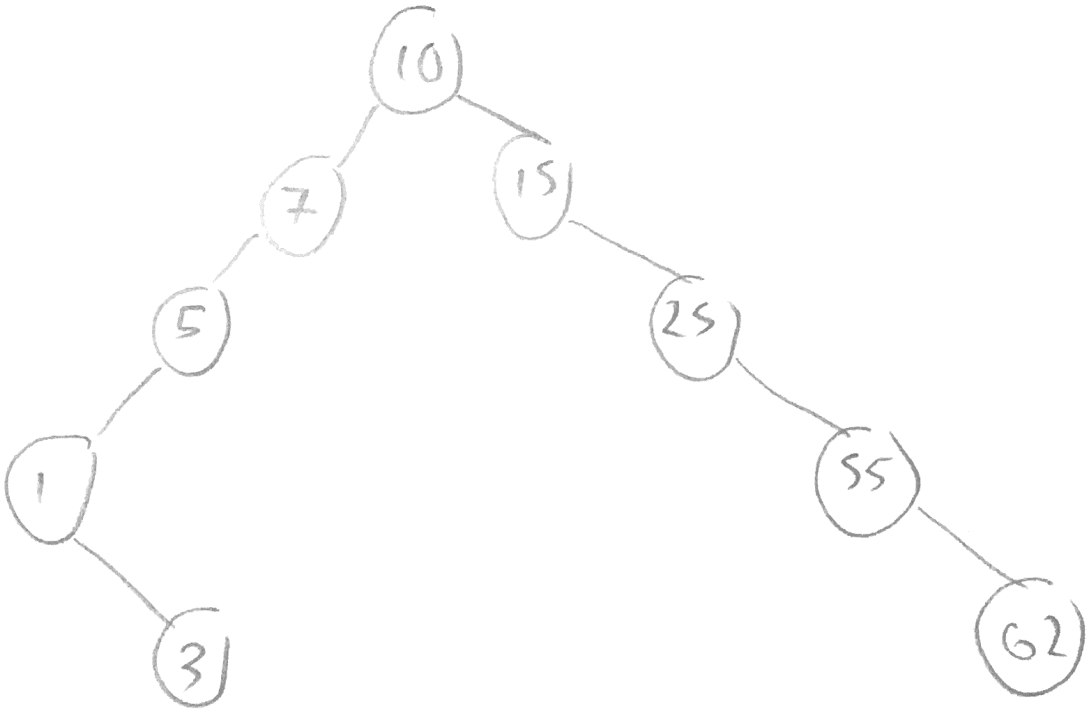
LR

Given the following list of data: 10, 15, 7, 5, 25, 55, 62, 1, 3
Build

3. Binary Tree



4. Binary Search Tree



10, 15, 7, 5, 25, 55, 62, 1, 3

5. AVL Tree

+ selection sort

- Find the smallest Item.
- swap to the First Item in unsorted array.

```
void selection_sort (int a[], int size){
    int i
    int index_smallest;
    for(i = 0; i < size-1; i++){
        int smallest = find_index(a, size, i);
        swap(&a[i], &a[index_smallest]);
    }
}
int find_index (int a[], int size, int start_index){
    int i;
    int index_min = start_index;
    for(i = start_index; i < size; i++){
        if(a[i] < a[index_min]){
            index_min = i;
        }
    return index_min;
```

8 2 2 7 9 31

8 7 22 9 31

8 7 9 22 31

7 8 9 22 31

n^2 Bubble

8 22 7 9 31 (smallest)

7 22 8 9 31

7 8 22 9 31

7 8 9 22 31

n^2 selection sort

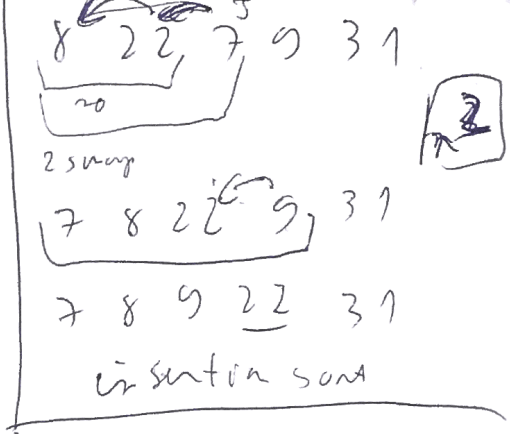
+ Bubble-sort

- Compare 1st pair.
- move through the list

```
void bubble_sort (int *a[], int size){
    int i, n;
    for(n=0; n < size-1; n++){
        for(i=0; i < size-1; i++){
            if(a[i] > a[i+1]){
                swap(&a[i], &a[i+1]);
}}}}
```

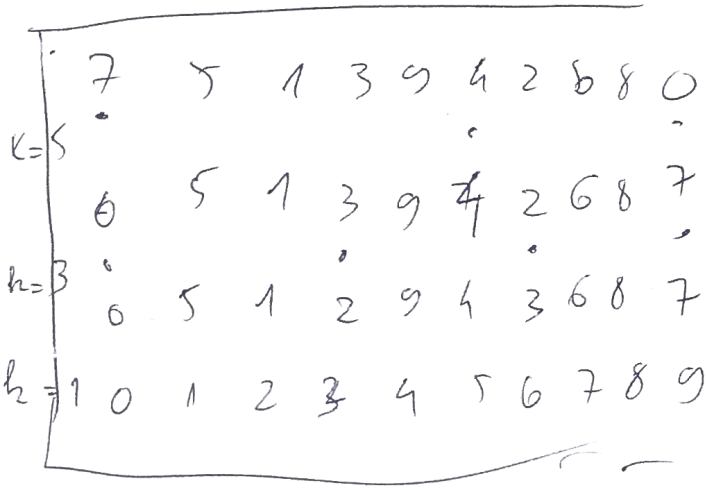
+ insertion sort

```
- void insertion_sort (int a[], int size){
    int i, j.
    for (i = 0; i < size; i++) {
        j = i;
        while (j-1 >=0 && a[j] < a[j-1]){
            swap ( & a[j], i], & a[j]
            j-- i } }
```



- Shell sort ($n^{3/2}$)

```
void shell_sort (int a[], int size)
{ int i, j, h;
    h = size/3+1.
    do {
        h--;
        for (i = h; i < size; i++) {
            j = i;
            while (a[j] < a[j-1]) {
                swap [& a[j-1], & a[j]) }
                j--;
            }
        }
    } while (h != 1);
}.
```



- Shell sort:
  + Initialize the value of h (gap)
  + Divide the list into smaller sub-list of equal interval h
  + Sort these sub-lists using insertion sort.
  + Decrease value of h and repeat the shell sort.

+ selection
- ~~Bubble~~ sort
  + Start from the first element of array, find the smallest element in the array and swap them
  + move to the next element and repeat the code.

+ Bubble sort.
  + Compare a pair of nodes starting from the first node of the array.
    - If the first node in the pair greater than the second node, swap them
    + else move on to the next element in the array and compare again until finish the array.
  + Repeat the code again until all element are sorted.

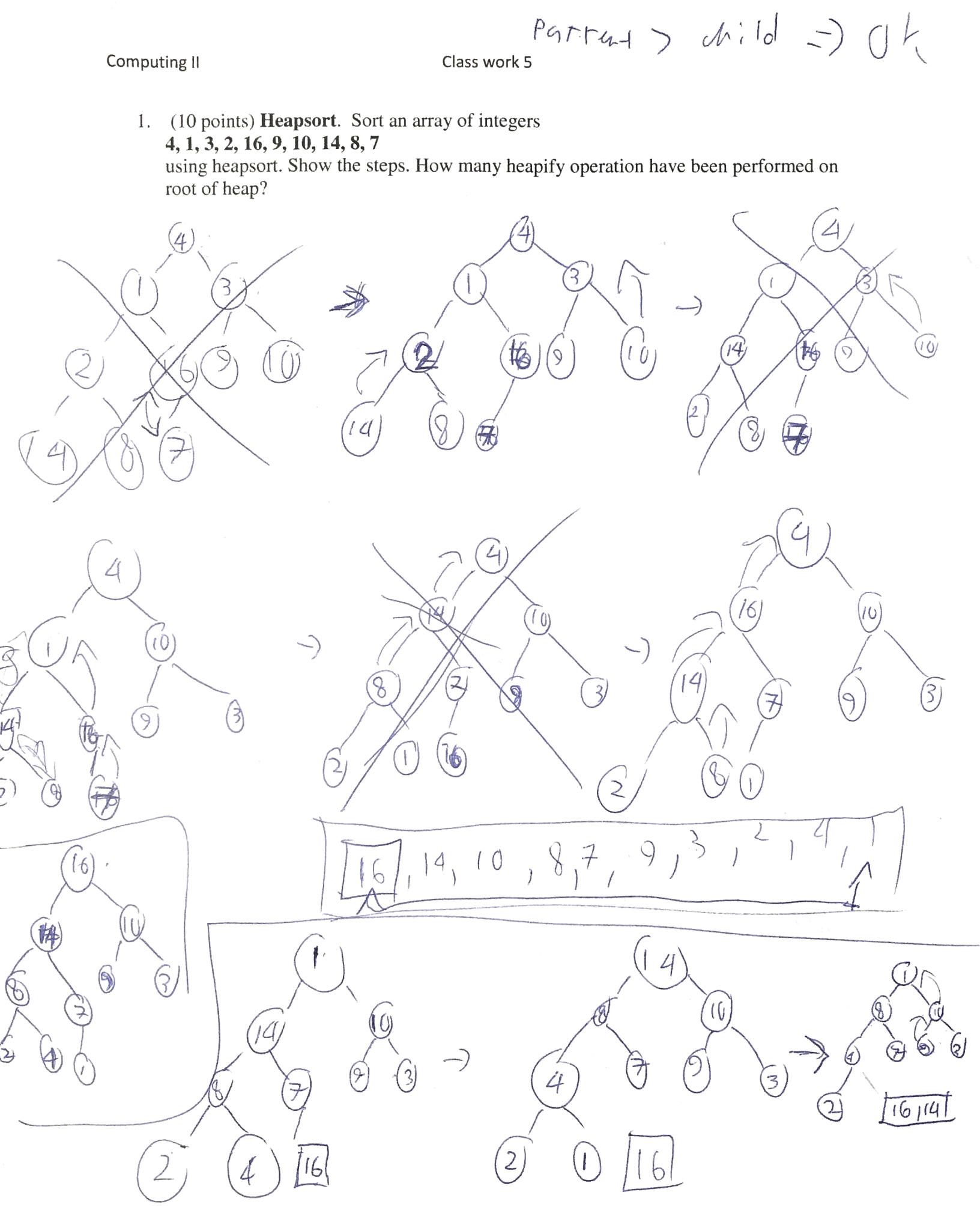
+ Insertion sort.
  + Start from first element of array, compare to the next element, swap if the second element less than first one.
  + Compare them to the next element and repeat until finish the array.

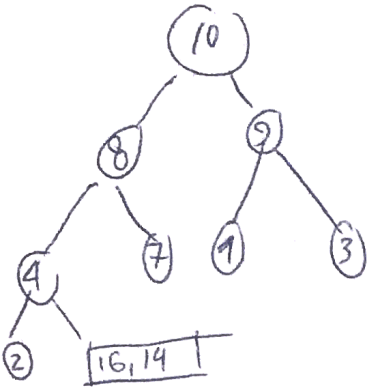
= Quick sort.

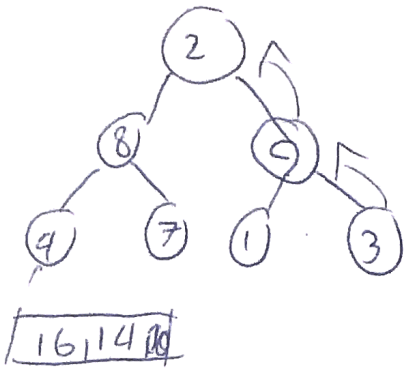
① + choose the highest index value has pivot
② + Take 2 pointer point to left and right of the list excluding pivot
③ + left point to the low index, right point to the high.
④ + while value at left is less than pivot move right
⑤ + while value at right is greater than pivot move left.
⑥ + if step 5 doesn't match, swap left & right
⑦ + if left ≥ right, the point where they met is new pivot

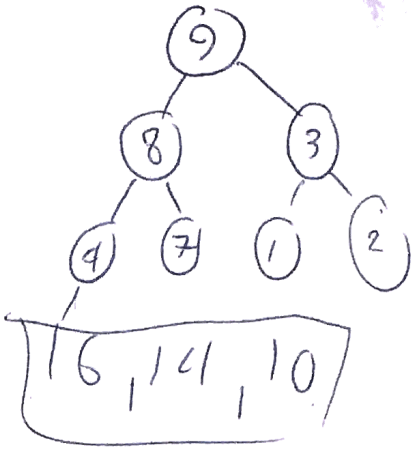
Parent > child => ok

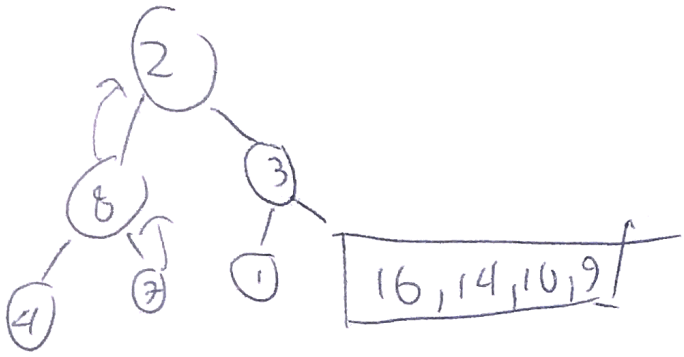
1. (10 points) **Heapsort**. Sort an array of integers
   4, 1, 3, 2, 16, 9, 10, 14, 8, 7
   using heapsort. Show the steps. How many heapify operation have been performed on root of heap?

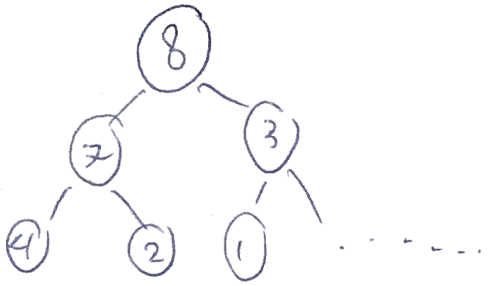


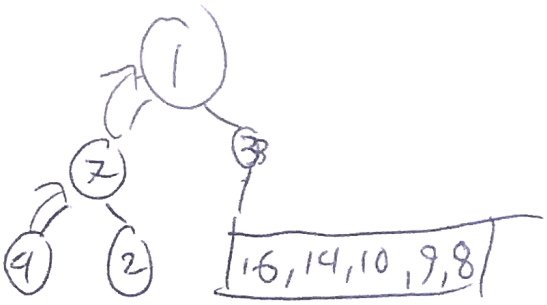

16,14


16,14,10


16,14,10


16,14,10,9




16,14,10,9,8


16,14,10,9,8


16,14,10,9,8,4,3,1