2020 2 학기
자료구조개론
기말고사

1. You will get +1 point for each correct answer, 0 for each unanswered question, and -0.3 for each wrong answer. Be careful when you guess.

2. Assume that each program includes proper header files such as stdio.h and math.h.

3. All the variables and arrays are properly initialized at the beginning.

Q1. Given static hash with open addressing, bucket size 13 and slot size 1, answer the questions

Index	Key
0	
1	Ⓐ
2	Ⓑ
3	
4	Ⓒ
5	
6	Ⓓ
7	
8	
9	
10	
11	
12	

Hash descriptions

- Static hashing
- Bucket size 13
- Slot size 1
- Open addressing
- Linear probe

Hash functions

H1(key)

⇨ **key % 13**

H2(key)

⇨ **(key * key) % 13**

Insert following keys in order

4
2
0
1
44
23
12
5
9
24
32

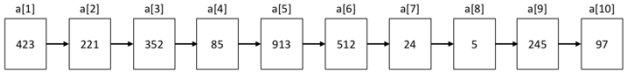
1. After insertion, what is the value at Ⓐ when hash function is H1 (if there's no value write NULL)
2. After insertion, what is the value at Ⓒ when hash function is H1 (if there's no value write NULL)
3. After insertion, what is the value at Ⓓ when hash function is H1 (if there's no value write NULL)
4. After insertion, what is the value at Ⓐ when hash function is H2 (if there's no value write NULL)
5. After insertion, what is the value at Ⓑ when hash function is H2 (if there's no value write NULL)
6. After insertion, what is the value at Ⓓ when hash function is H2 (if there's no value write NULL)

Q2. Read the following program, and answer the questions.

```
int sort(int a[], int link[], int d, int r, int n) {
    int front[r], rear[r];
    int i, bin, current, first, last;

    first = 1;

    for(i = 1; i < n; i++) link[i] = i+1;
    link[n] = 0;

    for(i = d-1; i >=0; i--) {
        for(bin = 0; bin < r; bin++) front[bin] = 0;

        for(current = first; current; current = link[current]){
            bin = digit(a[current], d - i, r);
            if(front[bin] == 0)     front[bin] = current;
            else    link[rear[bin]] = current;
            rear[bin] = current;
        }

        for(bin=0; !front[bin]; bin++);
        first=front[bin]; last=rear[bin];

        (A)

        for(bin++; bin < r; bin++) {
            if(front[bin]) {
                link[last] = front[bin]; last = rear[bin];
            }
        }
        link[last] = 0;

        (B)

    }
    return first;
}
```

```
int digit(int number, int i, int r)
{
    for (int div = 0; div < i; div++)
        number /= r;
    return number % r;
}
```

int a[] = {0, 423, 221, 352, 85, 913, 512, 24, 5, 245, 97};



7. What is the name of this sorting algorithm?

8. What is the time complexity of this algorithm?

9. In calling the function “sort”, what is the proper value of d and r to sort a[]? (<ex> d=-1, r=-1)

10. When i is d-1, what is the value of link[9] at (B)?

11. When i is d-1, what is the value of link[1] at (B)?

12. When i is d-1, what is the value of rear[9] at (B)?

13. When i is d-1, what is the value of the variable “bin” at (A)?

14. When i is d-2, what is the value of link[1] at (B)?

15. When i is 0, what is the value of a[first] at (B)?

16. when i is 0, what is the value of the variable “last” at (B)?

Q3. Read the following graph program, and answer the questions. **Assume "func" is called with <u>v = 0</u>.**

```
void func(int v, int cost[][MAX_VERTICES],
        int distance[], int n, short int found[])
{
      int i, u, w;
      for (i = 0; i < n; i++) {
            found[i] = FALSE;
            distance[i] = cost[v][i];
      }
      found[v] = TRUE;      distance[v] = 0;
      for (i = 0; i < n-2; i++) {
            u = choose(distance, n, found);
            found[u] = TRUE;
            for (w = 0; w < n; w++) {
                  if (!found[w]
                        && distance[u]+cost[u][w] < distance[w])
                        distance[w]= distance[u]+cost[u][w];
            }
      Ⓑ
      }
}
```

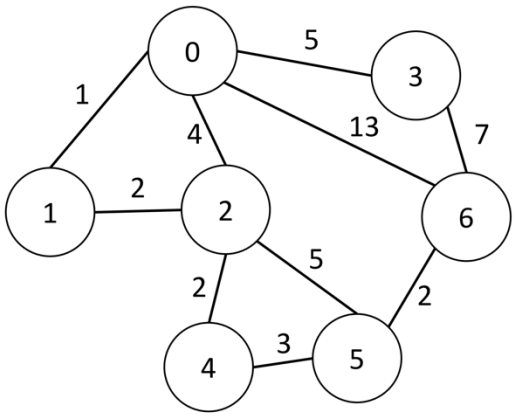
```
int choose(int distance[], int n, short int found[])
{
      /* finds the smallest distance not yet checked */
      int i, min, minpos;
      min = INT_MAX;
      minpos = -1;
      for (i = 0; i < n; i++) {
            if (distance[i] < min && !found[i]) {
                  min = distance[i];
                  minpos = i;
            }
      }
      return minpos;
}
```

MAX_VERTICES 7

```
int cost[][MAX_VERTICES] =
```

0,	1,	4,	5,	1000,	1000,	13
1,	0,	2,	1000,	1000,	1000,	1000
4,	2,	0,	1000,	2,	5,	1000
5,	1000,	1000,	0,	1000,	1000,	7
1000,	1000,	2,	1000,	0,	3,	1000
1000,	1000,	5,	1000,	3,	0,	2
13,	1000,	1000,	7,	1000,	2,	0

/* 1000 implies there is no connection between the vertices. */



17. What is the name of this graph algorithm?

18. When i is 0, what is the value of distance[1] at Ⓑ?

19. When i is 0, what is the value of distance[4] at Ⓑ?

20. When i is 1, what is the value of u at Ⓑ**?**

21. When i is 2, what is the value of u at Ⓑ**?**

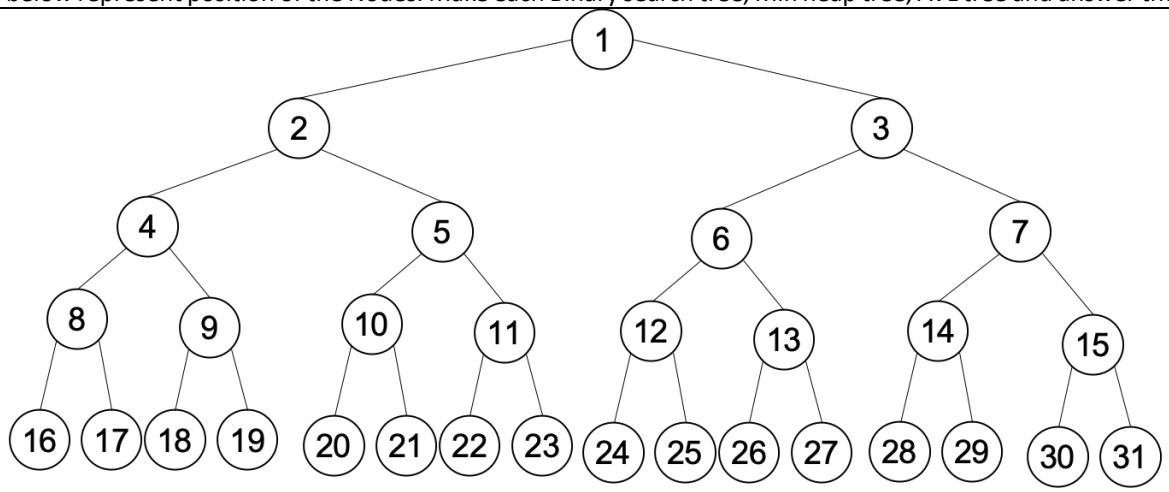
22. When i is 3, what is the value of u at Ⓑ**?**

23. When i is 4, what is the value of distance[5] at Ⓑ**?**

24. If the “func” is called with v = 4, what is the value of distance[5], when i is 0, at Ⓑ

25. If the “func” is called with v = 4, what is the value of distance[3], when i is 1, at Ⓑ**?**

Q4. Graph below represent position of the Nodes. Make each Binary search tree, Min heap tree, AVL tree and answer the questions



Binary search tree operations	Min heap tree operations (Heapify after every insert)
Insert 25 Insert 35 Insert 22 Insert 12 Insert 1 Insert 13 Insert 29 Insert 9 Insert 20 Insert 24 Insert 32 Insert 23 Insert 27	Insert 23 Insert 15 Insert 17 Insert 32 Insert 10 Insert 12 Insert 5 Insert 6 Insert 30 Insert 9 Insert 1

26. [BINARY_TREE]
    What is the value of Node 1 after all operations
    (Write NULL if there's no value)

27. [BINARY_TREE]
    What is the value of Node 5 after all operations
    (Write NULL if there's no value)

28. [BINARY_TREE]
    What is the value of Node 7 after all operations
    (Write NULL if there's no value)

29. [BINARY_TREE]
    What is the value of Node 17 after all operations
    (Write NULL if there's no value)

30. [BINARY_TREE]
    What is the value of Node 18 after all operations
    (Write NULL if there's no value)

31. [MIN_HEAP]
    What is the value of Node 1 after all operations
    (Write NULL if there's no value)

32. [MIN_HEAP]
    What is the value of Node 2 after all operations
    (Write NULL if there's no value)

33. [MIN_HEAP]
    What is the value of Node 6 after all operations
    (Write NULL if there's no value)

34. [MIN_HEAP]
    What is the value of Node 11 after all operations
    (Write NULL if there's no value)

35. [MIN_HEAP]
    What is the value of Node 12 after all operations
    (Write NULL if there's no value)

2020 2 학기
자료구조개론
기말고사

1. You will get +1 point for each correct answer, 0 for each unanswered question, and -0.3 for each wrong answer. Be careful when you guess.

2. Assume that each program includes proper header files such as stdio.h and math.h.

3. All the variables and arrays are properly initialized at the beginning.

Q1. Given static hash with open addressing, bucket size 13 and slot size 1, answer the questions

Index	Key		
0	0		0
1	1	Ⓐ	1
2	2	Ⓑ	12
3			4
4	4	Ⓒ	2
5	44		5
6	5	Ⓓ	9
7	32		24
8			
9	9		23
10	23		32
11	24		
12	12		44
	H1		H2

Hash descriptions

- Static hashing
- Bucket size 13
- Slot size 1
- Open addressing
- Linear probe

Hash functions

H1(key)
⇨ **key % 13**

H2(key)
⇨ **(key * key) % 13**

Insert following keys in order

4 16 → 3
2 4 → 4
0 0 → 0
1 1 → 1
44 44 1936 → 12
23 23 529 → 9
12 144 → 1
5 25 → 12
9 81 → 3
24 576 → 4
32 1024 → 10

1. After insertion, what is the value at Ⓐ when hash function is H1 (if there's no value write NULL) 1
2. After insertion, what is the value at Ⓒ when hash function is H1 (if there's no value write NULL) 4
3. After insertion, what is the value at Ⓓ when hash function is H1 (if there's no value write NULL) 5
4. After insertion, what is the value at Ⓐ when hash function is H2 (if there's no value write NULL) 1
5. After insertion, what is the value at Ⓑ when hash function is H2 (if there's no value write NULL) 12
6. After insertion, what is the value at Ⓓ when hash function is H2 (if there's no value write NULL) 9

Q1. 관련 필기 static hashing만 배웠기 알려줘고 해결되고

overflow를 대비 위해 openaddressing을 사용

강의에서는 2가지만 배움 ┌ Linear probing √ 이 문제에서 사용
└ Random probing → 교안 에서 보면서 대충해두기

linear probing → overflow 발생시 그 다음 slot을 확인하면서 저장해나감.

마지막 index에서 overflow 발생할경우 (H2에서 key=5 일 때, 여기 추가 들어옴)
처음으로 돌아가서 해쉬값 내려준 후 빈 slot이 있을때 2 slot에 입력

loading factor ┌ H1 = 11/13x1
└ H2 = 11/13x1

+ 시간되면 Hash function

1. Mid Square
2. division
→ 교안 보고공부해두기

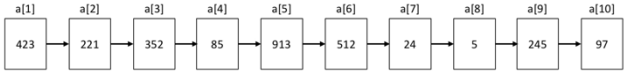
시험범위에 있는거 맞아..?

Q2. Read the following program, and answer the questions.

```
int sort(int a[], int link[], int d, int r, int n) {
    int front[r], rear[r];
    int i, bin, current, first, last;

    first = 1;

    for(i = 1; i < n; i++) link[i] = i+1;
    link[n] = 0;

    for(i = d-1; i >=0; i--) {
        for(bin = 0; bin < r; bin++) front[bin] = 0;

        for(current = first; current; current = link[current]){
            bin = digit(a[current], d - i, r);
            if(front[bin] == 0)     front[bin] = current;
            else    link[rear[bin]] = current;
            rear[bin] = current;
        }

        for(bin=0; !front[bin]; bin++);
        first=front[bin]; last=rear[bin];

        (A)

        for(bin++; bin < r; bin++) {
            if(front[bin]) {
                link[last] = front[bin]; last = rear[bin];
            }
        }
        link[last] = 0;

        (B)

    }
    return first;
}
```

```
int digit(int number, int i, int r)
{
    for (int div = 0; div < i; div++)
        number /= r;
    return number % r;
}
```

```
int a[] = {0, 423, 221, 352, 85, 913, 512, 24, 5, 245, 97};
```



7. What is the name of this sorting algorithm?

8. What is the time complexity of this algorithm?

9. In calling the function “sort”, what is the proper value of d and r to sort a[]? (<ex> d=-1, r=-1)

10. When i is d-1, what is the value of link[9] at (B)?

11. When i is d-1, what is the value of link[1] at (B)?

12. When i is d-1, what is the value of rear[9] at (B)?

13. When i is d-1, what is the value of the variable “bin” at (A)?

14. When i is d-2, what is the value of link[1] at (B)?

15. When i is 0, what is the value of a[first] at (B)?

16. when i is 0, what is the value of the variable “last” at (B)?

v=0 → 시작하는 node

Q3. Read the following graph program, and answer the questions. **Assume "func" is called with v = 0.**

```
void func(int v, int cost[][MAX_VERTICES],
        int distance[], int n, short int found[])
{
    int i, u, w;
    for (i = 0; i < n; i++) {
        found[i] = FALSE;
        distance[i] = cost[v][i];
    }
    found[v] = TRUE;     distance[v] = 0;
    for (i = 0; i < n-2; i++) {
        u = choose(distance, n, found);
        found[u] = TRUE;
        for (w = 0; w < n; w++) {
            if (!found[w]
                && distance[u]+cost[u][w] < distance[w])
                distance[w]= distance[u]+cost[u][w];
        }
    }
    B
}
```

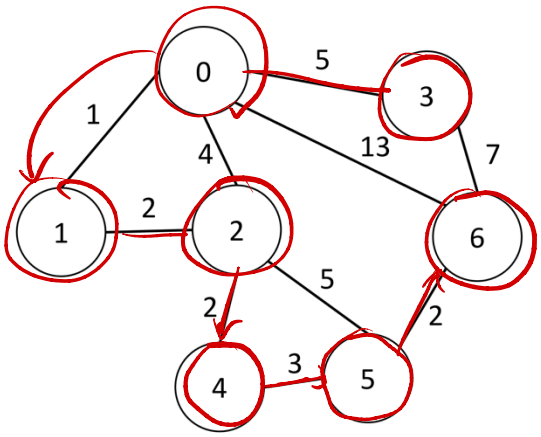
```
int choose(int distance[], int n, short int found[])
{
    /* finds the smallest distance not yet checked */
    int i, min, minpos;
    min = INT_MAX;
    minpos = -1;
    for (i = 0; i < n; i++) {
        if (distance[i] < min && !found[i]) {
            min = distance[i];
            minpos = i;
        }
    }
    return minpos;
}
```

MAX_VERTICES 7

int cost[][MAX_VERTICES] =

0,	1,	4,	5,	1000,	1000,	13
1,	0,	2,	1000,	1000,	1000,	1000
4,	2,	0,	1000,	2,	5,	1000
5,	1000,	1000,	0,	1000,	1000,	7
1000,	1000,	2,	1000,	0,	3,	1000
1000,	1000,	5,	1000,	3,	0,	2
13,	1000,	1000,	7,	1000,	2,	0

/* 1000 implies there is no connection between the vertices. */



distance

		[0]	[1]	[2]	[3]	[4]	[5]	[6]
v=1	i=0	0	1	3	5	1000	1000	13
v=2	i=1	0	1	3	5	5	8	13
v=3	i=2	0	1	3	5	5	8	12
v=4	i=3	0	1	3	5	5	8	12
v=5	i=4	0	1	3	5	5	8	10

17. What is the name of this graph algorithm?

다익스트라 최단경로

18. When i is 0, what is the value of distance[1] at Ⓑ?

1

19. When i is 0, what is the value of distance[4] at Ⓑ?

1000

20. When i is 1, what is the value of u at Ⓑ?

2

21. When i is 2, what is the value of u at Ⓑ?

3

22. When i is 3, what is the value of u at Ⓑ?

4

23. When i is 4, what is the value of distance[5] at Ⓑ?

8

24. If the "func" is called with v = 4, what is the value of distance[5], when i is 0, at Ⓑ

3

25. If the "func" is called with v = 4, what is the value of distance[3], when i is 1, at Ⓑ?

1000

v=4 distance

	[0]	[1]	[2] ✓	[3]	[4] ✓	[5] ✓	[6]
u=2 i=0	6	4	2	1000	0	3	1000
u=5 i=1	0	4	2	1000	0	3	5

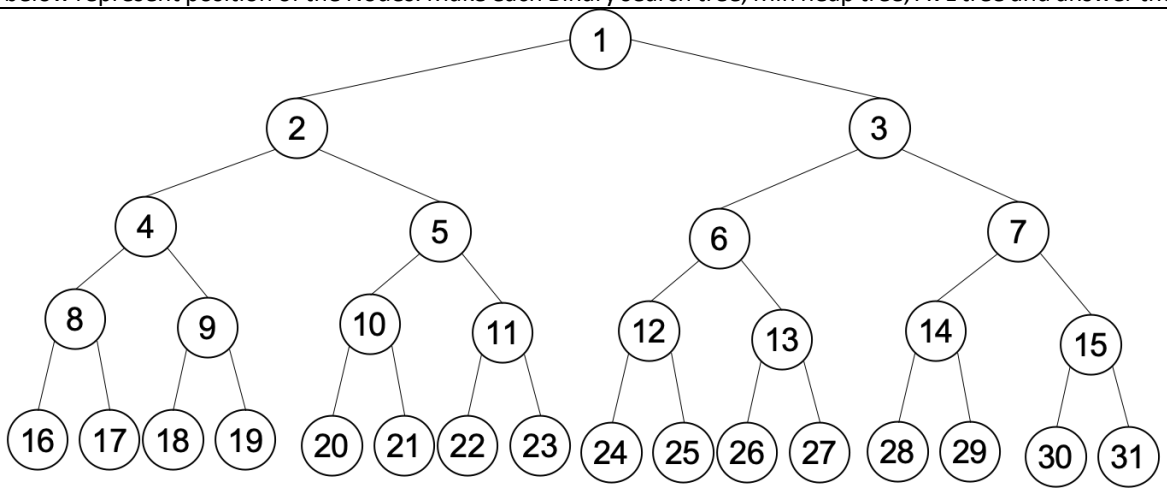
다익스트라 compile 해보기

i값이 변화함에 따라 방문(u)의 값을 찾고

distance 배열 갱신해주기 계속해서 돌아가야함.

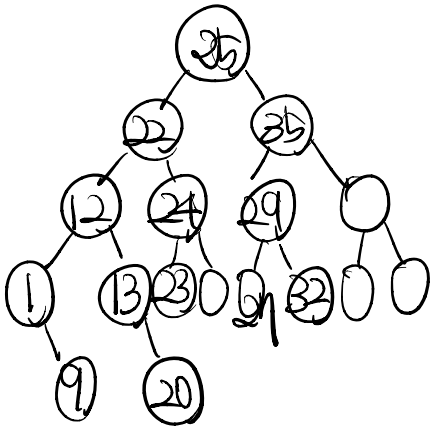
+ 방문한 노드 배열에서 거쳐가는지 바로바로 비교

Q4. Graph below represent position of the Nodes. Make each Binary search tree, Min heap tree, AVL tree and answer the questions



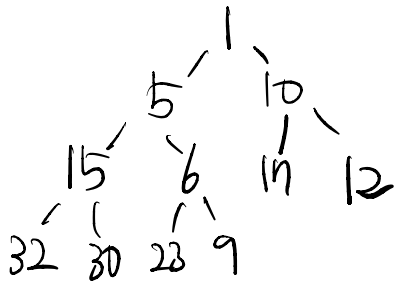
Binary search tree operations

Insert 25
Insert 35
Insert 22
Insert 12
Insert 1
Insert 13
Insert 29
Insert 9
Insert 20
Insert 24
Insert 32
Insert 23
Insert 27



Min heap tree operations
(Heapify after every insert)

Insert 23
Insert 15
Insert 17
Insert 32
Insert 10
Insert 12
Insert 5
Insert 6
Insert 30
Insert 9
Insert 1



26. [BINARY_TREE]
What is the value of Node 1 after all operations
(Write NULL if there's no value)

27. [BINARY_TREE]
What is the value of Node 5 after all operations
(Write NULL if there's no value)

28. [BINARY_TREE]
What is the value of Node 7 after all operations
(Write NULL if there's no value)

29. [BINARY_TREE]
What is the value of Node 17 after all operations
(Write NULL if there's no value)

30. [BINARY_TREE]
What is the value of Node 18 after all operations
(Write NULL if there's no value)

31. [MIN_HEAP]
What is the value of Node 1 after all operations
(Write NULL if there's no value)

32. [MIN_HEAP]
What is the value of Node 2 after all operations
(Write NULL if there's no value)

33. [MIN_HEAP]
What is the value of Node 6 after all operations
(Write NULL if there's no value)

34. [MIN_HEAP]
What is the value of Node 11 after all operations
(Write NULL if there's no value)

35. [MIN_HEAP]
What is the value of Node 12 after all operations
(Write NULL if there's no value)

2021 2 학기
자료구조개론
기말고사
문제지

1. You will get +1 point for each correct answer.

2. Assume that each program includes proper header files such as stdio.h and math.h.

3. All the variables and arrays are properly initialized by 0 at the beginning.

Q1. Read the following program, and answer the questions.

```
void Sort(int *arr, int len)
{
    int tmp = 0;
    for(int i = len - 1; i >= 0; i--){
        for(int j = 0; j < i; j++){
            if(Ⓐ){
                tmp = arr[j];
                Ⓑ
                arr[j + 1] = tmp;
            }
        }
    }
}
```

```
int main(void){
    int n = 10;

    int a[10] = {9, 11, 4, 2, 3, 1, 5 ,7 ,10, 6};

    // Sort ascending order
    Sort(a, n);

    for(int i = 0; i < n; i++){
        printf("%d₩n", a[i]);
    }
}
```

1. What is time complexity of the sort program? (n is the number of items in the list)
   ① Θ(1) ② Θ(n) ③ Θ($\log_2 n$)
   ④ Θ($n\log_2 n$) ⑤ Θ(n^2)

2. What is the name of this sort algorithm?
   ① Selection sort
   ② Insertion sort
   ③ Bubble sort
   ④ Quick sort
   ⑤ Merge sort

3. What is the value of a[4] after sort?
   ① 1 ② 3 ③ 5 ④ 7 ⑤ 9

4. What is the correct statement at Ⓐ?
   ① a[j+1] < a[j];
   ② a[j+1] > a[j];
   ③ a[j+1] == a[j];
   ④ a[j-1] > a[j];
   ⑤ a[j-1] < a[j];

5. What is the correct statement at Ⓑ?
   ① a[j] = a[j+1];
   ② a[j] = a[j-1];
   ③ a[j+1] = a[j];
   ④ a[j-1] = a[j];
   ⑤ a[j+1] = a[j-1];

Q2. Given static hash with open addressing, bucket size 17 and slot size 1, answer the questions

Index	Key
0	
1	Ⓐ
2	
3	
4	
5	Ⓑ
6	
7	
8	Ⓒ
9	
10	
11	
12	Ⓓ
13	
14	
15	
16	

Hash descriptions

- Static hashing
- Bucket size 17
- Slot size 1
- Open addressing
- Linear probe
- Empty slots have NULL

Hash functions

H1(key)

⇨ **key % 17**

H2(key)

⇨ **(key * key) % 17**

Insert following keys in order

17
24
29
0
3
4
87
54
15
35
12
48
65

6. After insertion, what is the value at Ⓐ when hash function is H1

① NULL ② 17 ③ 0 ④ 65 ⑤ 48

7. After insertion, what is the value at Ⓓ when hash function is H1

① NULL ② 4 ③ 48 ④ 35 ⑤ 29

8. After insertion, what is the value at Ⓑ when hash function is H1

① NULL ② 0 ③ 15 ④ 54 ⑤ 87

9. After insertion, what is the value at Ⓑ when hash function is H2

① NULL ② 0 ③ 17 ④ 15 ⑤ 48

10. After insertion, what is the value at Ⓓ when hash function is H2

① NULL ② 48 ③ 65 ④ 35 ⑤ 87

11. After insertion, what is the value at Ⓒ when hash function is H2

① NULL ② 24 ③ 29 ④ 17 ⑤ 54

Q3. Read the following program, and answer the questions.

```
#define MAX_SIZE 10

void fct1(int list1[], int list2[], int i, int m, int n)
{
	int j,k,t;
	j = m + 1;
	k = i;
	while(i <= m && j <= n) {
		if (list1[i] <= list1[j]){
			list2[k++] = list1[i++];
		}
		else{
			list2[k++] = list1[j++];
		}
	}

	Ⓐ

	if (i > m){
		for(t = j; t <= n; t++){
			list2[t] = list1[t];
		}
	}
	else{
		for(t = i; t <= m; t++){
			list2[k+t-i] = list1[t];
		}
	}
}

void fct2(int list1[], int list2[], int n, int s)
{
	int i ;
	int j ;

	for (i = 0; i <= n - 2 * s + 1; i += 2 * s){
		fct1(list1, list2, i, i + s - 1, i + 2 * s - 1);
	}

	Ⓑ

	if((i + s - 1) < n){
		fct1(list1, list2, i, i + s - 1, n);
	}
	else{
		for(j=i; j <= n; j++){
			list2[j] = list1[j];
		}
	}
}
```

```
void Sort(int a[], int n)
{
	int s = 1;
	int extra[MAX_SIZE];

	while (s<n) {
		fct2(a, extra, n, s);
		s *= 2;

		Ⓒ

		fct2(extra, a, n, s);
		s *= 2;

		Ⓓ

	}
}

int main(void)
{
	int arr[MAX_SIZE] = {13, 26, 41, 72, 23, 1, 0, 65, 32, 55};

	Sort(arr, MAX_SIZE - 1);
}
```

12. What is the name of this sorting algorithm?

① Merge Sort ④ LSD Radix Sort

② Heap Sort ⑤ List Sort

③ MSD Radix Sort

13. What is the time complexity of this algorithm?

① $\Theta(1)$ ② $\Theta(n)$ ③ $\Theta(\log_2 n)$

④ $\Theta(n\log_2 n)$ ⑤ $\Theta(n^2)$

14. How many times fct1 is called?

① 8 ② 9 ③ 10 ④ 11 ⑤ 12

15. How many times fct2 is called?

① 3 ② 4 ③ 5 ④ 6 ⑤ 7

16. When fct1 is called twice, what is the value of list1[4] at Ⓐ?

① 26 ② 23 ③ 1 ④ 0 ⑤ 55

17. When fct1 is called 5 times, what is the value of list2 [8] at Ⓐ?

① 32 ② 72 ③ 26 ④ 13 ⑤ 65

18. When fct2 is called for the first time, what is the value of list1[3] at Ⓑ?

① 26 ② 32 ③ 72 ④ 41 ⑤ 55

19. When fct2 is called 3 times, what is the value of list1[5] at Ⓑ?

① 41 ② 0 ③ 1 ④ 13 ⑤ 23

20. When s is 2, what is the value of a[3] at Ⓒ?

① 26 ② 32 ③ 41 ④ 72 ⑤ 13

21. When s is 4, what is the value of a[5] at Ⓓ?

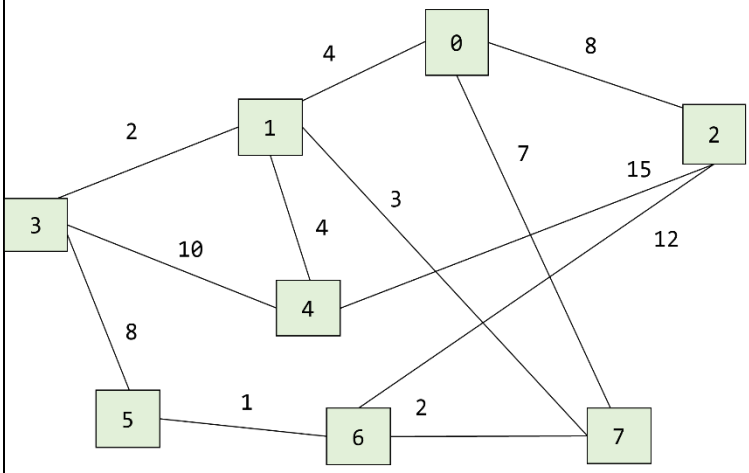
① 0 ② 23 ③ 1 ④ 55 ⑤ 65

Q4. Read the following graph program, and answer the questions. **Assume "func" is called with <u>v = 0</u>**.

```
#define VERY_LARGE_NUMBER 1000000
#define MAX_VERTICES 8

typedef struct __edge{
    int x;
    int y;
}edge;

typedef struct __treeData{
    edge T[MAX_VERTICES * MAX_VERTICES];
    int top;
}treeData;

void init_treeData(treeData *tree)
{
    edge tmp = {-1, -1};
    for(int i = 0; i < MAX_VERTICES * MAX_VERTICES; i++)
        tree->T[i] = tmp;
    tree->top = 0;
}
void fct(int graph[MAX_VERTICES][MAX_VERTICES]);

void push_to_T(treeData *tree, int x, int y)
{
    edge tmp;
    for(int i = 0; i < tree->top; i++){
        tmp = tree->T[i];
        if((tmp.x == x && tmp.y == y) || (tmp.x == y && tmp.y == x)){
            return;
        }
    }
    tree->T[tree->top].x = x;
    tree->T[tree->top].y = y;

    tree->top++;
}
```

```
void fct(int graph[MAX_VERTICES][MAX_VERTICES])
{
    int no_edge = 0;

    treeData tree;
    int TV[MAX_VERTICES] = {0};

    init_treeData(&tree);

    TV[0] = true;

    while(no_edge < MAX_VERTICES - 1) {
        int min = VERY_LARGE_NUMBER;
        edge tmp = {0, 0};

        for (int i = 0; i < MAX_VERTICES; i++) {
            if (TV[i]) {
                for (int j = 0; j < MAX_VERTICES; j++) {
                    if (!TV[j] && graph[i][j] != -1) {
                        if (min > graph[i][j]) {
                            min = graph[i][j];

                            push_to_T(&tree, i, j);

                            tmp.x = i;
                            tmp.y = j;

                            Ⓐ
                        }
                    }
                }
            }
        }
        TV[tmp.y] = true;
        no_edge++;
    }
}
```

```
int main(void)
{
/* -1 implies there is no connection between the vertices. */
    int graph[MAX_VERTICES][MAX_VERTICES] = {
        { 0,  4,  8, -1, -1, -1, -1,  7},
        { 4,  0, -1,  2,  4, -1, -1,  3},
        { 8, -1,  0, -1, 15, -1, 12, -1},
        {-1,  2, -1,  0, 10,  8, -1, -1},
        {-1,  4, 15, 10,  0, -1, -1, -1},
        {-1, -1, -1,  8, -1,  0,  1, -1},
        {-1, -1, 12, -1, -1,  1,  0,  2},
        { 7,  3, -1, -1, -1, -1,  2,  0}};

    fct(graph);

    return 0;
}
```



22. What is the name of this graph algorithm?

① Kruskal's Algorithm

② Prim's Algorithm

③ Dijkstra's Algorithm

④ Bellman-Ford Algorithm

⑤ Floyd-Warshall Algorithm

23. Find the edge that is in the MST made by the above code.

① (4, 2) ② (1, 4) ③ (5, 3) ④ (3, 4) ⑤ (0, 7)

24. What is tree.top after the fct ended?

① 6 ② 7 ③ 8 ④ 9 ⑤ 10

25. Find the edge that cannot be in the tree.T when tree.top is 3 at Ⓐ.

① (0, 1) ② (0, 7) ③ (1, 4) ④ (1, 7) ⑤ (6, 5)

26. Find the edge that cannot be in the tree.T when tree.top is 6 at Ⓐ.

① (0, 1) ② (0, 2) ③ (1, 7) ④ (6, 5) ⑤ (3, 4)

27. Suppose that node 1 and node 7 are not connected.
Find the edge that is in the MST made by the above code and given assumption.

① (0, 1) ② (2, 6) ③ (5, 3) ④ (3, 4) ⑤ (4, 2)

28. Suppose that node 1 and node 7 are not connected.
What is tree.top after the fct ended?

① 6 ② 7 ③ 8 ④ 9 ⑤ 10

29. Suppose that node 1 and node 7 are not connected.

Find the edge that cannot be in the tree.T when tree.top is 3 at Ⓐ.

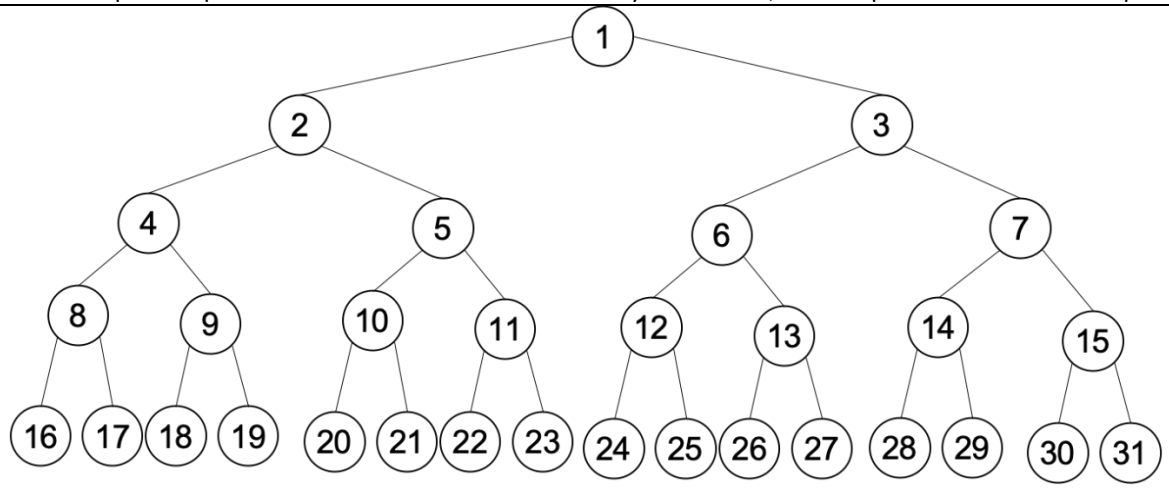
① (0, 1) ② (0, 2) ③ (1, 3) ④ (0, 7) ⑤ (2, 4)

30. Suppose that node 1 and node 7 are not connected.

Find the edge that cannot be in the tree.T when tree.top is 6 at Ⓐ.

① (0, 1) ② (7, 6) ③ (3, 4) ④ (6, 5) ⑤ (1, 4)

Q5. Graph below represent position of the Nodes. Make each Binary search tree, Min heap tree and answer the questions



Binary search tree operations	Max heap tree operations (Heapify after every insert)
Insert 17 Insert 5 Insert 24 Insert 15 Insert 30 Insert 34 Insert 2 Insert 1 Insert 19 Insert 22 Insert 28 Insert 37 Insert 11 Insert 13 Insert 4	Insert 17 Insert 5 Insert 24 Insert 15 Insert 30 Insert 34 Insert 2 Insert 1 Insert 19 Insert 22 Insert 28 Insert 37

31. [BINARY_TREE]
What is the value of Node 7 after all operations
① NULL ② 34 ③ 13 ④ 30 ⑤ 22

32. [BINARY_TREE]
What is the value of Node 15 after all operations
① NULL ② 34 ③ 13 ④ 30 ⑤ 22

33. [BINARY_TREE]
What is the value of Node 19 after all operations
① NULL ② 34 ③ 13 ④ 30 ⑤ 22

34. [MAX_HEAP]
What is the value of Node 1 after all operations
① NULL ② 28 ③ 34 ④ 30 ⑤ 37

35. [MAX_HEAP]
What is the value of Node 5 after all operations
① NULL ② 24 ③ 19 ④ 30 ⑤ 22

36. [MAX_HEAP]
What is the value of Node 9 after all operations
① NULL ② 5 ③ 15 ④ 22 ⑤ 17

2021 2 학기

자료구조개론

기말고사

문제지

1. You will get +1 point for each correct answer.

2. Assume that each program includes proper header files such as stdio.h and math.h.

3. All the variables and arrays are properly initialized by 0 at the beginning.

Q1. Read the following program, and answer the questions.

```
void Sort(int *arr, int len)
{
    int tmp = 0;
    for(int i = len - 1; i >= 0; i--){
        for(int j = 0; j < i; j++){
            if(Ⓐ){
                tmp = arr[j];
                Ⓑ
                arr[j + 1] = tmp;
            }
        }
    }
}
```

```
int main(void){
    int n = 10;
    int a[10] = {9, 11, 4, 2, 3, 1, 5, 7, 10, 6};

    // Sort ascending order
    Sort(a, n);

    for(int i = 0; i < n; i++){
        printf("%d₩n", a[i]);
    }
}
```

1. What is time complexity of the sort program? (n is the number of items in the list)
   ① Θ(1) ② Θ(n) ③ Θ($\log_2 n$)
   ④ Θ($n\log_2 n$) ⑤ Θ(n^2)

2. What is the name of this sort algorithm?
   ① Selection sort
   ② Insertion sort
   ③ Bubble sort
   ④ Quick sort
   ⑤ Merge sort

3. What is the value of a[4] after sort?
   ① 1 ② 3 ③ 5 ④ 7 ⑤ 9

4. What is the correct statement at Ⓐ?
   ① a[j+1] < a[j];
   ② a[j+1] > a[j];
   ③ a[j+1] == a[j];
   ④ a[j-1] > a[j];
   ⑤ a[j-1] < a[j];

5. What is the correct statement at Ⓑ?
   ① a[j] = a[j+1];
   ② a[j] = a[j-1];
   ③ a[j+1] = a[j];
   ④ a[j-1] = a[j];
   ⑤ a[j+1] = a[j-1];

Q2. Given static hash with open addressing, bucket size 17 and slot size 1, answer the questions

Index	Key
0	17
1	0 Ⓐ
2	87
3	3
4	4
5	54 Ⓑ 15
6	35
7	24
8	Ⓒ 29
9	3
10	54
11	12
12	29 Ⓓ 48
13	12 65
14	48
15	15 24
16	65 4

Hash descriptions

- Static hashing
- Bucket size 17
- Slot size 1
- Open addressing
- Linear probe
- Empty slots have NULL

Hash functions

H1(key)
⇨ **key % 17**

H2(key)
⇨ **(key * key) % 17**

random 아님 ㅠㅠ

Insert following keys in order

		H1	H2
17	289	0	0
24	576	7	15
29	841	12	8
0	0	0	0
3	9	3	9
4	16	4	16
87	7569	2	4
54	2916	3	9
15	225	15	4
35	1225	1	1
12	144	12	8
48	2304	14	9
65	4225	14	9

6. After insertion, what is the value at Ⓐ when hash function is H1

① NULL ② 17 ③ 0 ④ 65 ⑤ 48

7. After insertion, what is the value at Ⓓ when hash function is H1

① NULL ② 4 ③ 48 ④ 35 ⑤ 29

8. After insertion, what is the value at Ⓑ when hash function is H1

① NULL ② 0 ③ 15 ④ 54 ⑤ 87

9. After insertion, what is the value at Ⓑ when hash function is H2

① NULL ② 0 ③ 17 ④ 15 ⑤ 48

10. After insertion, what is the value at Ⓓ when hash function is H2

① NULL ② 48 ③ 65 ④ 35 ⑤ 87

11. After insertion, what is the value at Ⓒ when hash function is H2

① NULL ② 24 ③ 29 ④ 17 ⑤ 54

merge sort 문제

Q3. Read the following program, and answer the questions.

```
#define MAX_SIZE 10

void fct1(int list1[], int list2[], int i, int m, int n)
{
    int j,k,t;
    j = m + 1;
    k = i;
    while(i <= m && j <= n) {
        if (list1[i] <= list1[j]){
            list2[k++] = list1[i++];
        }
        else{
            list2[k++] = list1[j++];
        }
    }

    Ⓐ

    if (i > m){
        for(t = j; t <= n; t++){
            list2[t] = list1[t];
        }
    }
    else{
        for(t = i; t <= m; t++){
            list2[k+t-i] = list1[t];
        }
    }
}

void fct2(int list1[], int list2[], int n, int s)
{
    int i ;
    int j ;

    for (i = 0; i <= n - 2 * s + 1; i += 2 * s){
        fct1(list1, list2, i, i + s - 1, i + 2 * s - 1);
    }

    Ⓑ

    if((i + s - 1) < n){
        fct1(list1, list2, i, i + s - 1, n);
    }
    else{
        for(j=i; j <= n; j++){
            list2[j] = list1[j];
        }
    }
}
```

```
void Sort(int a[], int n)
{
    int s = 1;
    int extra[MAX_SIZE];

    while (s<n) {
        fct2(a, extra, n, s);
        s *= 2;

        Ⓒ

        fct2(extra, a, n, s);
        s *= 2;

        Ⓓ

    }
}

int main(void)
{
    int arr[MAX_SIZE] = {13, 26, 41, 72, 23, 1, 0, 65, 32, 55};

    Sort(arr, MAX_SIZE - 1);
}
```

★ 이 문제는 제외 (32번)

12. What is the name of this sorting algorithm?

① Merge Sort ④ LSD Radix Sort

② Heap Sort ⑤ List Sort

③ MSD Radix Sort

13. What is the time complexity of this algorithm?

① Θ(1) ② Θ(n) ③ Θ($\log_2 n$)

④ Θ($n\log_2 n$) ⑤ Θ(n^2)

14. How many times fct1 is called?

① 8 ② 9 ③ 10 ④ 11 ⑤ 12

5+2+1+1

15. How many times fct2 is called?

① 3 ② 4 ③ 5 ④ 6 ⑤ 7

16. When fct1 is called twice, what is the value of list1[4] at (A)?

① 26 ② 23 ③ 1 ④ 0 ⑤ 55

17. When fct1 is called 5 times, what is the value of list2 [8] at (A)?

① 32 ② 72 ③ 26 ④ 13 ⑤ 65

18. When fct2 is called for the first time, what is the value of list1[3] at (B)?

① 26 ② 32 ③ 72 ④ 41 ⑤ 55

19. When fct2 is called 3 times, what is the value of list1[5] at (B)?

① 41 ② 0 ③ 1 ④ 13 ⑤ 23

20. When s is 2, what is the value of a[3] at (C)?

① 26 ② 32 ③ 41 ④ 72 ⑤ 13

21. When s is 4, what is the value of a[5] at (D)?

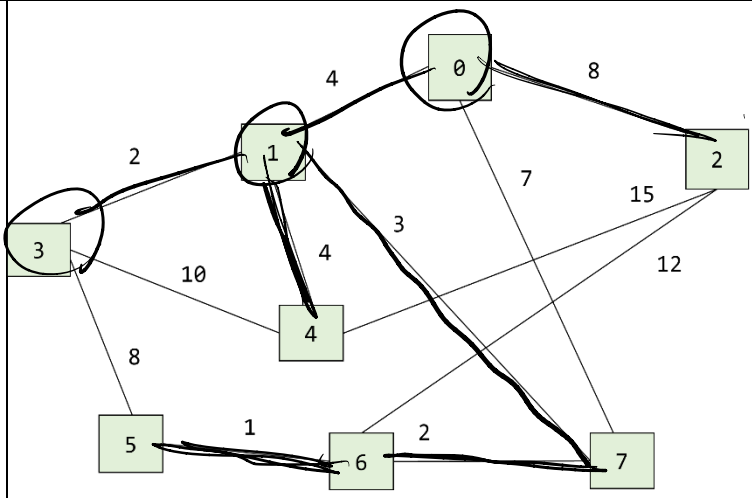
① 0 ② 23 ③ 1 ④ 55 ⑤ 65

Q4. Read the following graph program, and answer the questions. **Assume "func" is called with v = 0**.

```
#define VERY_LARGE_NUMBER 1000000
#define MAX_VERTICES 8

typedef struct __edge{
    int x;
    int y;
}edge;

typedef struct __treeData{
    edge T[MAX_VERTICES * MAX_VERTICES];
    int top;
}treeData;

void init_treeData(treeData *tree)
{
    edge tmp = {-1, -1};
    for(int i = 0; i < MAX_VERTICES * MAX_VERTICES; i++)
        tree->T[i] = tmp;
    tree->top = 0;
}
void fct(int graph[MAX_VERTICES][MAX_VERTICES]);

void push_to_T(treeData *tree, int x, int y)
{
    edge tmp;
    for(int i = 0; i < tree->top; i++){
        tmp = tree->T[i];
        if((tmp.x == x && tmp.y == y) || (tmp.x == y && tmp.y == x)){
            return;
        }
    }
    tree->T[tree->top].x = x;
    tree->T[tree->top].y = y;

    tree->top++;
}
```

```
void fct(int graph[MAX_VERTICES][MAX_VERTICES])
{
    int no_edge = 0;

    treeData tree;
    int TV[MAX_VERTICES] = {0};

    init_treeData(&tree);

    TV[0] = true;

    while(no_edge < MAX_VERTICES - 1) {
        int min = VERY_LARGE_NUMBER;
        edge tmp = {0, 0};

        for (int i = 0; i < MAX_VERTICES; i++) {
            if (TV[i]) {
                for (int j = 0; j < MAX_VERTICES; j++) {
                    if (!TV[j] && graph[i][j] != -1) {
                        if (min > graph[i][j]) {
                            min = graph[i][j];

                            push_to_T(&tree, i, j);

                            tmp.x = i;
                            tmp.y = j;

                            Ⓐ
                        }
                    }
                }
            }
        }
        TV[tmp.y] = true;
        no_edge++;
    }
}
```

```
int main(void)
{
/* -1 implies there is no connection between the vertices. */
    int graph[MAX_VERTICES][MAX_VERTICES] = {
        { 0,  4,  8, -1, -1, -1, -1,  7},
        { 4,  0, -1,  2,  4, -1, -1,  3},
        { 8, -1,  0, -1, 15, -1, 12, -1},
        {-1,  2, -1,  0, 10,  8, -1, -1},
        {-1,  4, 15, 10,  0, -1, -1, -1},
        {-1, -1, -1,  8, -1,  0,  1, -1},
        {-1, -1, 12, -1, -1,  1,  0,  2},
        { 7,  3, -1, -1, -1, -1,  2,  0}};

    fct(graph);

    return 0;
}
```



(0,1) → (1,3) → (1,7) → (7,6)

(6,5) → (1,4) → (0,2)

22. What is the name of this graph algorithm?

① Kruskal's Algorithm

② Prim's Algorithm

③ Dijkstra's Algorithm

④ Bellman-Ford Algorithm

⑤ Floyd-Warshall Algorithm

23. Find the edge that is in the MST made by the above code.

① (4, 2) ② (1, 4) ③ (5, 3) ④ (3, 4) ⑤ (0, 7)

24. What is tree.top after the fct ended?

① 6 ② 7 ③ 8 ④ 9 ⑤ 10

25. Find the edge that cannot be in the tree.T when tree.top is 3 at Ⓐ.

① (0, 1) ② (0, 7) ③ (1, 4) ④ (1, 7) ⑤ (6, 5)

26. Find the edge that cannot be in the tree.T when tree.top is 6 at Ⓐ.

① (0, 1) ② (0, 2) ③ (1, 7) ④ (6, 5) ⑤ (3, 4)

27. Suppose that node 1 and node 7 are not connected.
Find the edge that is in the MST made by the above code and given assumption.

① (0, 1) ② (2, 6) ③ (5, 3) ④ (3, 4) ⑤ (4, 2)

28. Suppose that node 1 and node 7 are not connected.
What is tree.top after the fct ended?

① 6 ② 7 ③ 8 ④ 9 ⑤ 10

29. Suppose that node 1 and node 7 are not connected.

Find the edge that cannot be in the tree.T when tree.top is 3 at Ⓐ.

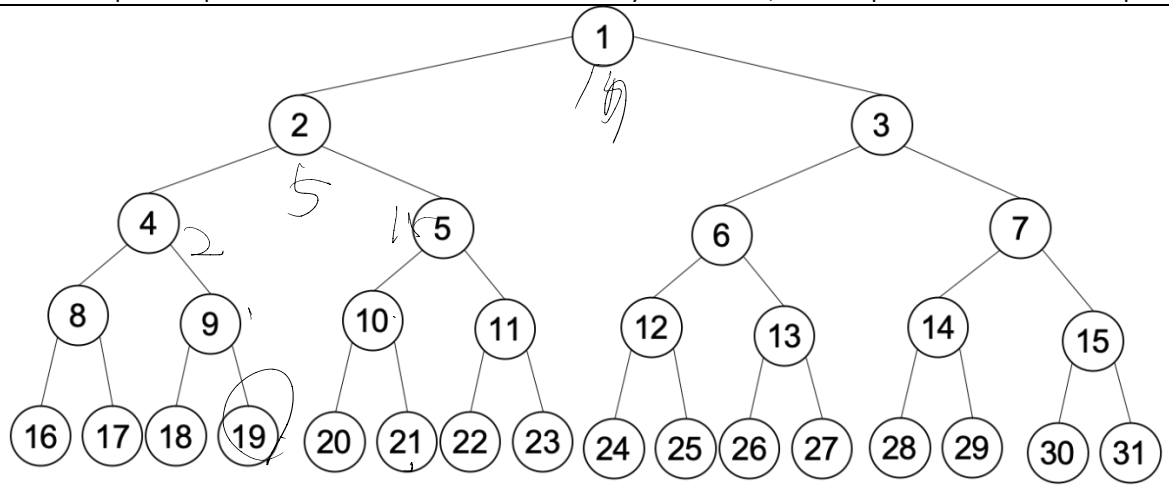
① (0, 1) ② (0, 2) ③ (1, 3) ④ (0, 7) ⑤ (2, 4)

30. Suppose that node 1 and node 7 are not connected.

Find the edge that cannot be in the tree.T when tree.top is 6 at Ⓐ.

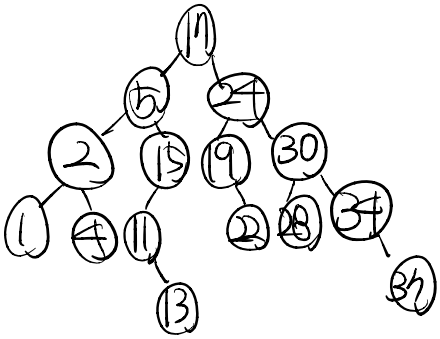
① (0, 1) ② (7, 6) ③ (3, 4) ④ (6, 5) ⑤ (1, 4)

Q5. Graph below represent position of the Nodes. Make each Binary search tree, Min heap tree and answer the questions



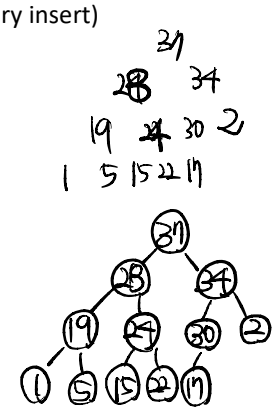
Binary search tree operations

Insert 17
Insert 5
Insert 24
Insert 15
Insert 30
Insert 34
Insert 2
Insert 1
Insert 19
Insert 22
Insert 28
Insert 37
Insert 11
Insert 13
Insert 4



Max heap tree operations
(Heapify after every insert)

Insert 17
Insert 5
Insert 24
Insert 15
Insert 30
Insert 34
Insert 2
Insert 1
Insert 19
Insert 22
Insert 28
Insert 37



31. [BINARY_TREE]
What is the value of Node 7 after all operations
① NULL ② 34 ③ 13 ④ 30 ⑤ 22

32. [BINARY_TREE]
What is the value of Node 15 after all operations
① NULL ② 34 ③ 13 ④ 30 ⑤ 22

33. [BINARY_TREE]
What is the value of Node 19 after all operations
① NULL ② 34 ③ 13 ④ 30 ⑤ 22

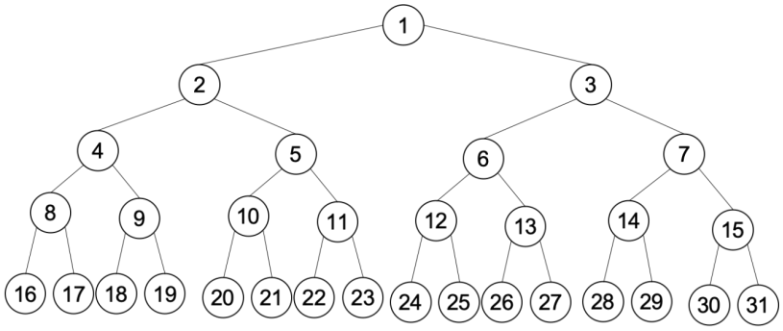
34. [MAX_HEAP]
What is the value of Node 1 after all operations
① NULL ② 28 ③ 34 ④ 30 ⑤ 37

35. [MAX_HEAP]
What is the value of Node 5 after all operations
① NULL ② 24 ③ 19 ④ 30 ⑤ 22

36. [MAX_HEAP]
What is the value of Node 9 after all operations
① NULL ② 5 ③ 15 ④ 22 ⑤ 17

2022 2학기 자료구조개론 기말고사 문제지

The following figure represents the position of nodes in a binary tree. Answer the questions (1~10).



[BST] Suppose that we have a binary search tree, and insert keys as the following order:

17, 5, 24, 15, 30, 34, 2, 1, 19, 22, 28, 37

1. Where is key 22 located after all the insertions?

(1) 6 (2) 7 (3) 12 (4) 13 (5) 14

2. What is the key of Node 7 after all the insertions?

(1) 22 (2) 24 (3) 28 (4) 30 (5) 34

Now, we delete keys as the following order: 24, 30, 15

Here, assume that we replace the deleted node with the smallest element in the right subtree when we delete a node with two child nodes.

3. What is the key of Node 3 after all the deletions?

(1) 19 (2) 22 (3) 28 (4) 34 (5) 37

4. Where is key 34 located after all the deletions?

(1) 3 (2) 6 (3) 7 (4) 14 (5) 15

[Max heap] Suppose that we have a max heap, and insert keys as the following order:

17, 5, 24, 15, 30, 34, 2, 1, 19, 22, 28, 37

5. What is the key of Node 7 after all the insertions?

(1) 1 (2) 2 (3) 5 (4) 15 (5) 17

6. What is the key of Node 9 after all the insertions?

(1) 1 (2) 2 (3) 5 (4) 15 (5) 17

7. What is the key of Node 12 after all the insertions?

(1) 1 (2) 2 (3) 5 (4) 15 (5) 17

Now, we call pop() three times.

8. What is the key of Node 5 after three pop()s?

(1) 1 (2) 2 (3) 5 (4) 15 (5) 17

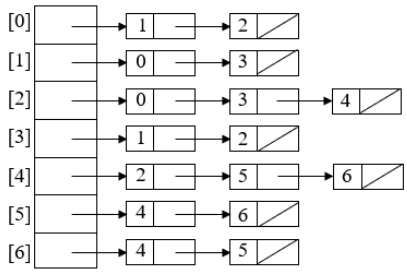
9. What is the key of Node 6 after three pop()s?

(1) 1 (2) 2 (3) 5 (4) 15 (5) 17

10. What is the key of Node 7 after three pop()s?

(1) 1 (2) 2 (3) 5 (4) 15 (5) 17

A graph is given by the following adjacency lists, and *low(u)* is defined as follows where *dfn(u)* is the depth first number. Suppose that we construct a depth first spanning tree with node 4 as the root, and answer the following questions (11~20).



$$low(u) = \min\{dfn(u),$$
$$\min\{low(w) \mid w \text{ is a child of } u\},$$
$$\min\{dfn(w) \mid (u,w) \text{ is a back edge}\}\}$$

11. How many back edges are there in the resulting depth first spanning tree?
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

12. What is dfn(0)? Note that dfn(0) is the depth first number of node 0.
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

13. What is dfn(3)?
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

14. What is dfn(4)?
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

15. What is low(0)?
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

16. What is low(2)?
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

17. What is low(3)?
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

18. What is low(4)?
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

19. What is low(5)?
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

20. What is low(6)?
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

Consider a static hash table with bucket size 17 and slot size 1. Bucket index starts from 0 to 16. When overflow occurs, it is handled by linear probing. The hash function is given by key%17, and the keys are inserted in the following order: 17, 24, 29, 0, 3, 4, 20, 18, 15, 19, 12

21. Which key is in bucket[0]?
(1) 0 (2) 17 (3) 18 (4) 20 (5) None

22. Which key is in bucket[2]?
(1) 17 (2) 18 (3) 19 (4) 20 (5) None

23. Which key is in bucket[5]?
(1) 17 (2) 18 (3) 19 (4) 20 (5) None

24. Which key is in bucket[14]?
(1) 12 (2) 17 (3) 18 (4) 15 (5) None

25. How many probes are required to search key 18? If you can find the key at bucket[key%17], the number of probes is 1.
(1) 1 (2) 2 (3) 3 (4) 4 (5) 5

26. How many probes are required to search key 19?
(1) 1 (2) 2 (3) 3 (4) 4 (5) 5

27. How many probes are required to find out that key 21 is not in the hash table?
(1) 1 (2) 2 (3) 3 (4) 4 (5) 5

The following program is for quicksort. Answer the questions.

```c
#include <stdio.h>
#define SWAP(x,y,t) ((t) = (x), (x) = (y), (y) = (t))

int count = 0;

void quicksort(int[], int, int);

int main() {
    int a[10] = {26, 5, 37, 1, 61, 11, 59, 15, 48, 19};
    int i;

    quicksort(a, 0, 9);

    return 0;
}

void  quicksort (int a[], int left, int right)
{
    int pivot, i, j, temp;

    count ++;
    printf("count = %d: left = %d, right = %d\n", count, left, right);
    for(i = 0; i< 10; i++) printf("%d ", a[i]);
    printf("\n");

    if (left < right) {
        i = left; j = right + 1;
        pivot = a[left];
        do {
            do  i++;  while (a[i] < pivot); /*&& i<right*/
            do  j--;  while (a[j] > pivot); /*&& j>left */
            if (i < j) SWAP(a[i], a[j], temp);
        } while (i < j);
        SWAP (a[left], a[j], temp);
        quicksort(a, left, j - 1);
        quicksort(a, j + 1, right);
    }
}
```

28. What is the value of count when the program is done?
(1) 11 (2) 12 (3) 13 (4) 14 (5) 15

29. When count = 2, what is the value of right at line 22?
(1) 1 (2) 2 (3) 3 (4) 4 (5) 5

30. When count = 2, what is the value of a[0] at line 22?
(1) 1 (2) 5 (3) 11 (4) 15 (5) 19

31. When count = 3, what is the value of right at line 22?
(1) 1 (2) 2 (3) 3 (4) 4 (5) 5

32. When count = 3, what is the value of a[0] at line 22?
(1) 1 (2) 5 (3) 11 (4) 15 (5) 19

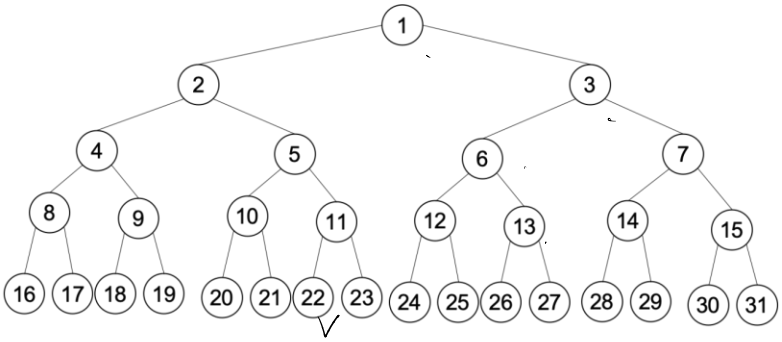
33. When count = 6, what is the value of right at line 22?
(1) 1 (2) 2 (3) 3 (4) 4 (5) 5

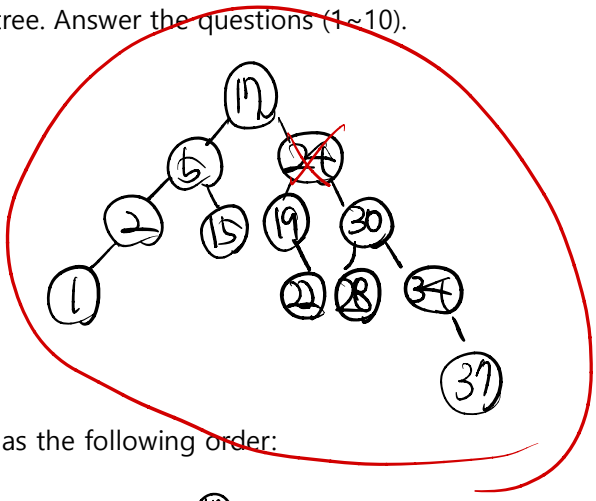
34. When count = 6, what is the value of a[3] at line 22?
(1) 1 (2) 5 (3) 11 (4) 15 (5) 19

/* end of exam. */

2022 2학기 자료구조개론 기말고사 문제지

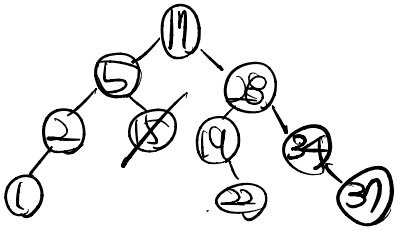
The following figure represents the position of nodes in a binary tree. Answer the questions (1~10).





[BST] Suppose that we have a binary search tree, and insert keys as the following order:

17, 5, 24, 15, 30, 34, 2, 1, 19, 22, 28, 37



1. Where is key 22 located after all the insertions?
   (1) 6 (2) 7 (3) 12 (4) 13 (5) 14
2. What is the key of Node 7 after all the insertions?
   (1) 22 (2) 24 (3) 28 (4) 30 (5) 34

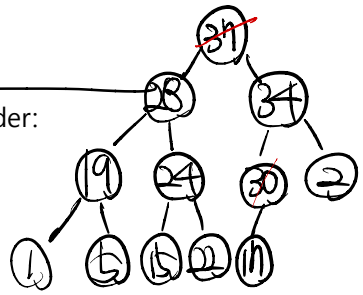
Now, we delete keys as the following order: 24, 30, 15

Here, assume that we replace the deleted node with the smallest element in the right subtree when we delete a node with two child nodes.

3. What is the key of Node 3 after all the deletions?
   (1) 19 (2) 22 (3) 28 (4) 34 (5) 37
4. Where is key 34 located after all the deletions?
   (1) 3 (2) 6 (3) 7 (4) 14 (5) 15

[Max heap] Suppose that we have a max heap, and insert keys as the following order:

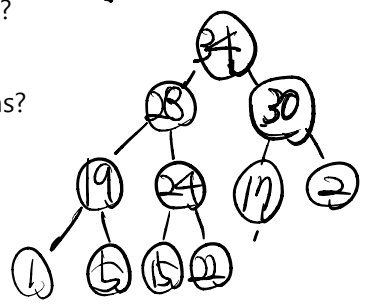
17, 5, 24, 15, 30, 34, 2, 1, 19, 22, 28, 37

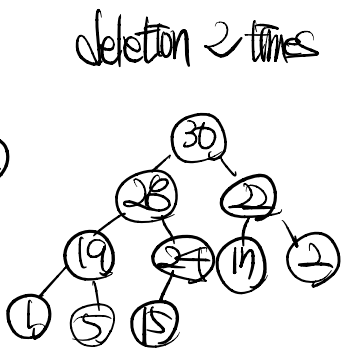


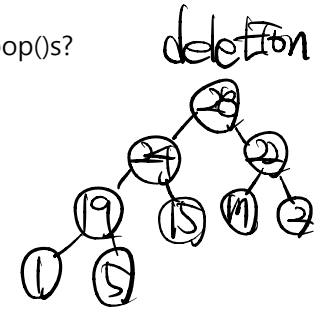
5. What is the key of Node 7 after all the insertions?
   (1) 1 (2) 2 (3) 5 (4) 15 (5) 17
6. What is the key of Node 9 after all the insertions?
   (1) 1 (2) 2 (3) 5 (4) 15 (5) 17
7. What is the key of Node 12 after all the insertions?
   (1) 1 (2) 2 (3) 5 (4) 15 (5) 17

Now, we call pop() three times.

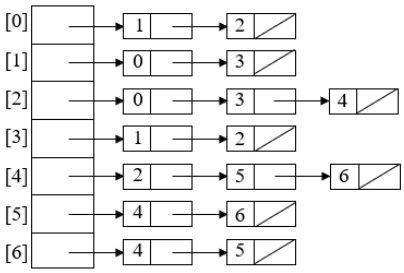
8. What is the key of Node 5 after three pop()s?
   (1) 1 (2) 2 (3) 5 (4) 15 (5) 17
9. What is the key of Node 6 after three pop()s?
   (1) 1 (2) 2 (3) 5 (4) 15 (5) 17
10. What is the key of Node 7 after three pop()s?
    (1) 1 (2) 2 (3) 5 (4) 15 (5) 17



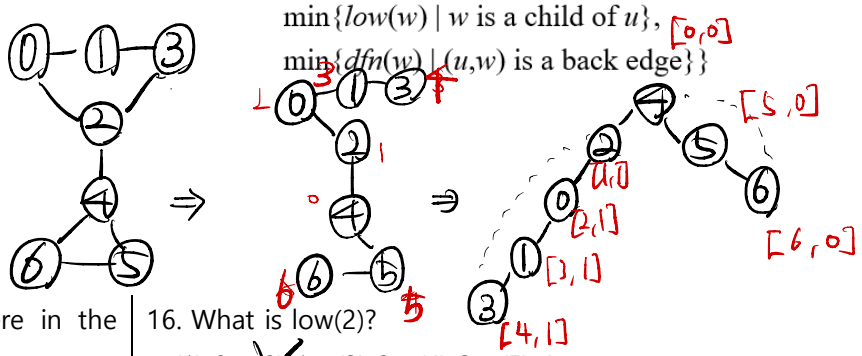




A graph is given by the following adjacency lists, and *low(u)* is defined as follows where *dfn(u)* is the depth first number. Suppose that we construct a depth first spanning tree with node 4 as the root, and answer the following questions (11~20).



$$low(u) = \min\{dfn(u),$$
$$\min\{low(w) \mid w \text{ is a child of } u\},$$
$$\min\{dfn(w) \mid (u,w) \text{ is a back edge}\}\}$$



11. How many back edges are there in the resulting depth first spanning tree?
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

12. What is dfn(0)? Note that dfn(0) is the depth first number of node 0.
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

13. What is dfn(3)?
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

14. What is dfn(4)?
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

15. What is low(0)?
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

16. What is low(2)?
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

17. What is low(3)?
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

18. What is low(4)?
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

19. What is low(5)?
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

20. What is low(6)?
(1) 0 (2) 1 (3) 2 (4) 3 (5) 4

Consider a static hash table with bucket size 17 and slot size 1. Bucket index starts from 0 to 16. When overflow occurs, it is handled by linear probing. The hash function is given by key%17, and the keys are inserted in the following order: 17, 24, 29, 0, 3, 4, 20, 18, 15, 19, 12

21. Which key is in bucket[0]?
(1) 0 (2) 17 (3) 18 (4) 20 (5) None

22. Which key is in bucket[2]?
(1) 17 (2) 18 (3) 19 (4) 20 (5) None

23. Which key is in bucket[5]?
(1) 17 (2) 18 (3) 19 (4) 20 (5) None

24. Which key is in bucket[14]?
(1) 12 (2) 17 (3) 18 (4) 15 (5) None

25. How many probes are required to search key 18? If you can find the key at bucket[key%17], the number of probes is 1.
(1) 1 (2) 2 (3) 3 (4) 4 (5) 5

26. How many probes are required to search key 19?
(1) 1 (2) 2 (3) 3 (4) 4 (5) 5

27. How many probes are required to find out that key 21 is not in the hash table?
(1) 1 (2) 2 (3) 3 (4) 4 (5) 5

The following program is for quicksort. Answer the questions.

```
1  #include <stdio.h>
2  #define SWAP(x,y,t) ((t) = (x), (x) = (y), (y) = (t))
3
4  int count = 0;
5
6  void quicksort(int[], int, int);
7
8  int main() {
9      int a[10] = {26, 5, 37, 1, 61, 11, 59, 15, 48, 19};
10     int i;
11
12     quicksort(a, 0, 9);
13
14     return 0;
15 }
16
17 void  quicksort (int a[], int left, int right)
18 {
19     int pivot, i, j, temp;
20
21     count ++;
22     printf("count = %d: left = %d, right = %d\n", count, left, right);
23     for(i = 0; i< 10; i++) printf("%d ", a[i]);
24     printf("\n");
25
26     if (left < right) {
27         i = left; j = right + 1;
28         pivot = a[left];
29         do {
30             do  i++;  while (a[i] < pivot); /*&& i<right*/
31             do  j--;  while (a[j] > pivot); /*&& j>left */
32             if (i < j) SWAP(a[i], a[j], temp);
33         } while (i < j);
34         SWAP (a[left], a[j], temp);
35         quicksort(a, left, j - 1);
36         quicksort(a, j + 1, right);
37     }
38 }
```

28. What is the value of count when the program is done?
    (1) 11 (2) 12 (3) 13 (4) 14 (5) 15

29. When count = 2, what is the value of right at line 22?
    (1) 1 (2) 2 (3) 3 (4) 4 (5) 5

30. When count = 2, what is the value of a[0] at line 22?
    (1) 1 (2) 5 (3) 11 (4) 15 (5) 19

31. When count = 3, what is the value of right at line 22?
    (1) 1 (2) 2 (3) 3 (4) 4 (5) 5

32. When count = 3, what is the value of a[0] at line 22?
    (1) 1 (2) 5 (3) 11 (4) 15 (5) 19

33. When count = 6, what is the value of right at line 22?
    (1) 1 (2) 2 (3) 3 (4) 4 (5) 5

34. When count = 6, what is the value of a[3] at line 22?
    (1) 1 (2) 5 (3) 11 (4) 15 (5) 19

/* end of exam. */