Problem B

오엑스를 맞춰보자 1

제한 시간 : 1초, 메모리 제한 : 128MB, 문제유형 : 이론

여러분은 두 달간 자료구조 이론에 대해 공부했었다. 이제 다음 오엑스 문제들에 대해 답할 시간이 되었다!

우리가 만들 프로그램은, 문제 번호가 입력으로 제시될 때,
그 문제에 해당하는 답을 출력하여 주는 프로그램을 완성해 보는 것이다.
이 때, 답이 O라면 1을, 답이 X라면 2를 출력하기로 하자.

현재 미완성된 프로그램은 다음과 같다. (6번 줄에서, 배열의 빈칸에 정답을 채워 넣은 소스 코드를 제출하면 된다.)

```
1  #include <iostream>
2  #define PROBLEM_NUM_MAX 10 // 최대 문제 수
3  using namespace std;
4
5  // 1번 문제의 정답은 answer[0]에, 2번 문제의 정답은 answer[1]에, ... 기입한다.
6  int answer[PROBLEM_NUM_MAX] = { x  x  x  x  x  x  x  x  x  x };
7
8  int main(void) {
9      int problem_number;
10     cin >> problem_number;
11     for (int i = 1; i <= PROBLEM_NUM_MAX; i++) {
12         if (problem_number == i) {
13             cout << answer[i - 1] << endl;
14             break;
15         }
16     }
17     return 0;
18 }
19
```

아래 10문제를 보고, O, X 여부를 판단하여 위 소스 코드를 완성하여라. 예를 들어, 1번 문제의 답이 O라면, 입력값이 1일 때 출력값은 1이다. 만약 3번 문제의 답이 X라면, 입력값이 3일 때 출력값은 2이다.

1번 문제. 큐는 먼저 들어간 것이 나중에 나오는 구조이다. ( )

2번 문제. 연결 리스트, 스택, 큐는 선형적인 자료구조이고, 트리와 그래프는 비선형적인 자료구조이다. ( )

3번 문제. 트리는 계층적인 자료구조라고도 하며 데이터베이스(DB)에서 널리 쓰인다. ( )

4번 문제. 이진 트리가 되려면, 루트 노드를 중심으로 둘로 나눠는 두 개의 서브 트리도 이진 트리여야 하고,그 서브 트리의 모든 서브 트리도 이진 트리이어야 한다. ( )

5번 문제. 모든 레벨이 꽉 찬 이진 트리를 포화 이진 트리(Full binary tree)라고 한다. ( )

6번 문제. 다음 수식 7+4*2-1은 중위 표기법을 이용한 것이다. ( )

7번 문제. 다음 수식 12+7*은 후위 표기법을 이용한 것이다. ( )

8번 문제. 12+7- 과 1+2-7은 실질적으로는 동일한 수식이다. ( )

9번 문제. 수식 트리의 구성과정에서 피연산자는 무조건 스택으로 옮긴다. ( )

10번 문제. C++ STL에서, set의 경우 중복된 원소는 한 번만 넣을 수 있으며 여러 번 insert(삽입 연산)를 한다고 해도 한 번만 들어가게 된다. ( )

Problem C

오엑스를 맞춰보자 2

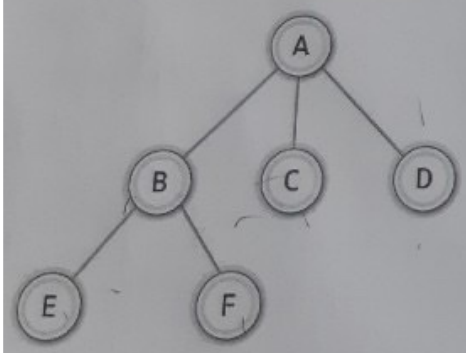
제한 시간 : 1초, 메모리 제한 : 128MB, 문제유형 : 이론

여러분은 두 달간 자료구조 이론에 대해 공부했었다. 이제 다음 오엑스 문제들에 대해 답할 시간이 되었다!
우리가 만들 프로그램은, 문제 번호가 입력으로 제시될 때,
그 문제에 해당하는 답을 출력하여 주는 프로그램을 완성해 보는 것이다.
이 때, 답이 O라면 1을, 답이 X라면 2를 출력하기로 하자.
현재 미완성된 프로그램은 다음과 같다. (6번 줄에서, 배열의 빈칸에 정답을 채워 넣은 소스 코드를 제출하면 된다.)

```
1  #include <iostream>
2  #define PROBLEM_NUM_MAX 10 // 최대 문제 수
3  using namespace std;
4
5  // 1번 문제의 정답은 answer[0]에, 2번 문제의 정답은 answer[1]에, ... 기입한다.
6  int answer[PROBLEM_NUM_MAX] = { ㅇ  ㅇ  ㅇ  ㅇ  ㅇ  ㅇ  ㅇ  ㅇ  ㅇ  ㅇ };
7
8  int main(void) {
9      int problem_number;
10     cin >> problem_number;
11     for (int i = 1; i <= PROBLEM_NUM_MAX; i++) {
12         if (problem_number == i) {
13             cout << answer[i - 1] << endl;
14             break;
15         }
16     }
17     return 0;
18 }
19
```

아래 10문제를 보고, 위 소스 코드를 완성하여라.

[1~4번 문제 : 다음 그림을 보고 물음에 답하라]



1번 문제. Root Node는 A에 해당한다. (ㅇ)
2번 문제. Edge는 총 6개이다. (ㅇ)
3번 문제. Leaf Node는 B,C,D,E,F에 해당한다. (~~X~~) E,F,C,D
4번 문제. 노드 B, C, D는 부모 노드가 같으므로, 서로가 서로에게 sibling node이다. (ㅇ)

[첫 번째 문제 : 다음 그림을 보고 물음에 답하라]

5번 문제. 위 그래프는 트리에 해당한다. ( )
6번 문제. 위 그래프는 방향 완전 그래프이다. ( )
7번 문제. 다음 그래프에서 V(G) = {동수, 지율, 민석, 지민, 수정, 정희} 이고,
E(G) = {<동수, 지율>, <지율,민석>, <민석, 지민>, <지민, 동수>, <민석, 정희>, <수정, 민석>, <정희, 수정>} 으로 나타낼 수 있다. ( )
8번 문제. 그래프는 인접 행렬과 인접 리스트로 나타낼 수 있다. ( )
9번 문제. 그래프의 탐색 방법 중 BFS는 깊이 우선 탐색으로 Stack을 사용한다. ( )
10번 문제. 최소 비용 신장 트리(MST)를 구성하기 위한 대표적인 알고리즘으로 Prim의 알고리즘과 Kruskal의 알고리즘이 있다. 이는 MST 내에 사이클이 존재하면 안되는 성질에 기반한다. ( )

Input

문제 번호가 입력으로 제시된다. 1번 문제의 경우 1, 2번 문제의 경우 2가 제시된다. 문제 번호는 1부터 5 사이의 정수이다.

Output

각 문제에 대한 답을 하나의 정수로 출력한다. 예를 들어, 1번 문제의 답이 3이라면 3을 출력한다.
아래 예시는 1번 문제의 답이 3일 때와, 2번 문제의 답이 5일 때를 나타낸 것이다.
예제는 채점하지 않으며 자신이 맞다고 생각하는 답을 출력하면 된다.

예제 1

Sample Input	Sample Output
1	3

예제 2

Sample Input	Sample Output
2	5

Problem F

오엑스를 맞춰보자 3

제한 시간 : 1초, 메모리 제한 : 128MB, 문제유형 : 이론

여러분은 두 달간 자료구조 이론에 대해 공부했었다. 이제 다음 오엑스 문제들에 대해 답할 시간이 되었다!

우리가 만들 프로그램은, 문제 번호가 입력으로 제시될 때,
그 문제에 해당하는 답을 출력하여 주는 프로그램을 완성해 보는 것이다.
이 때, 답이 O라면 1을, 답이 X라면 2를 출력하기로 하자.

현재 미완성된 프로그램은 다음과 같다. (6번 줄에서, 배열의 빈칸에 정답을 채워 넣은 소스 코드를 제출하면 된다.)

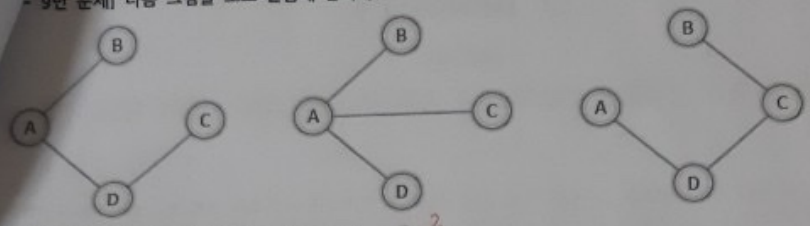
```
1  #include <iostream>
2  #define PROBLEM_NUM_MAX 10 // 최대 문제 수
3  using namespace std;
4
5  // 1번 문제의 정답은 answer[0]에, 2번 문제의 정답은 answer[1]에, ... 기입한다.
6  int answer[PROBLEM_NUM_MAX] = { ?  ?  ?  ?  ?  ?  ?  ?  ?  ? };
7
8  int main(void) {
9      int problem_number;
10     cin >> problem_number;
11     for (int i = 1; i <= PROBLEM_NUM_MAX; i++) {
12         if (problem_number == i) {
13             cout << answer[i - 1] << endl;
14             break;
15         }
16     }
17     return 0;
18 }
19
```

아래 10문제를 보고, O, X 여부를 판단하여 위 소스 코드를 완성하여라. 예를 들어, 1번 문제의 답이 O라면, 입력값이 1일 때 출력값은 1이다. 만약 3번 문제의 답이 X라면, 입력값이 3일 때 출력값은 2이다.

1번 문제. 스택은 먼저 들어간 것이 나중에 나오는 구조이다. (1)
2번 문제. 연결 리스트에서는 정렬 기능, 삽입 기능을 구현할 수 없다. (2)
3번 문제. 연결 리스트에서, 새 노드를 추가할 때 리스트의 머리에 저장할 경우 포인터 변수 tail이 불필요하다. (1)
4번 문제. 리스트는 연결 리스트만을 의미한다. (2)
5번 문제. O(1)은 상수형 빅-오라고 하며 데이터 수에 상관없이 연산횟수가 고정임을 의미한다. (1)
6번 문제. T(n) = $n^2 + \log n + n + 3$일 때 이를 Big O로 나타내면 O($\log n$)이다. (2)

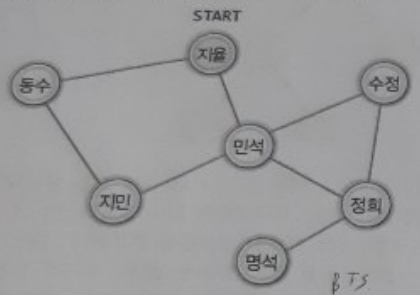
2

- 9번 문제] 다음 그림을 보고 물음에 답하라.



7번 문제. 위 그림들은 모두 방향 그래프이다. (2 2)
8번 문제. 위 그림들은 모두 트리이다. (/ /)
9번 문제. 위 그림들은 모두 신장 트리(Spanning Tree)에 해당한다. (/ /)

[10번 문제] 다음 그림을 보고 물음에 답하라.



BFS

10번 문제. 지율을 기준으로 너비 우선 탐색을 시행하면 조건에 따라 여러 가지 순서가 존재할 수 있으며, 지율, 동수, 민석, 지민, 수정, 정희, 명석의 순으로 탐색할 수 있다. (/ /)

Input

문제 번호가 입력으로 제시된다. 1번 문제의 경우 1, 2번 문제의 경우 2가 제시된다. 문제 번호는 1부터 5 사이의 정수이다.

Output

각 문제에 대한 답을 하나의 정수로 출력한다. 예를 들어, 1번 문제의 답이 3이라면 3을 출력한다.
아래 예시는 1번 문제의 답이 3일 때와, 2번 문제의 답이 5일 때를 나타낸 것이다.
문제는 채점하지 않으며 자신이 맞다고 생각하는 답을 출력하면 된다.

제 1

ample Input	Sample Output
	3

2

ple Input	Sample Output
	5

Problem I

객관식을 맞춰보자 1

제한 시간 : 1초, 메모리 제한 : 128MB, 문제유형 : 이론

문제
여러분은 두 달간 자료구조 이론에 대해 공부했었다. 이제 다음 오엑스 문제들에 대해 답할 시간이 되었다!

우리가 만들 프로그램은, 문제 번호가 입력으로 제시될 때,
그 문제에 해당하는 답을 출력하여 주는 프로그램을 완성해 보는 것이다.
이 때, 답이 O라면 1을, 답이 X라면 2를 출력하기로 하자.

현재 미완성된 프로그램은 다음과 같다. (6번 줄에서, 배열의 빈칸에 정답을 채워 넣은 소스 코드를 제출하면 된다.)

```
1   #include <iostream>
2   #define PROBLEM_NUM_MAX 10 // 최대 문제 수
3   using namespace std;
4
5   // 1번 문제의 정답은 answer[0]에, 2번 문제의 정답은 answer[1]에, ... 기입한다.
6   int answer[PROBLEM_NUM_MAX] = { ㅇ  ㅇ  ㅇ  ㅇ  ㅇ  ㅇ  ㅇ  ㅇ  ㅇ  ㅇ };
7
8   int main(void) {
9       int problem_number;
10      cin >> problem_number;
11      for (int i = 1; i <= PROBLEM_NUM_MAX; i++) {
12          if (problem_number == i) {
13              cout << answer[i - 1] << endl;
14              break;
15          }
16      }
17      return 0;
18  }
19
```

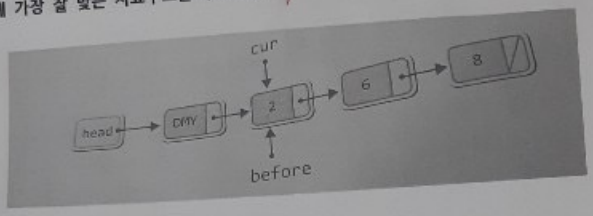
1번 문제. 다음 중 틀린 것은? (2)

(1) 이진 탐색 알고리즘(binary search algorithm)보다 순차 탐색 알고리즘(sequential search algorithm)이 더 느리다.
(2) ADT(Abstract Data Type)는 추상 자료형이라고 하며, 구체적인 기능의 완성과정을 언급한 것이다.
(3) 리스트는 배열을 통해서도 구현할 수 있다.
(4) 연결 리스트에서 더미 노드(Dummy Node)를 사용하는 이유는 노드의 추가, 삭제 및 조회 과정을 일관되게 구성할 수 있기 때문이다.
(5) 큐는 선입선출(FIFO)의 성질을 지닌다. 즉, 먼저 들어온 값이 먼저 나가게 된다.

2번 문제. 다음 보기 중 가장 적절하지 않은 것은? (4)

(1) 재귀함수의 문제점 중 하나는 중복된 계산이 시행될 수 있다는 점이다.
(2) T(n) = 3n + 2일 때, O(n) = n이다.
(3) 일반적인 이진 탐색 알고리즘의 시간 복잡도는 O(log n)이다.

리스트를 자료형과 포인터 변수를 이용하여 구현하는 것의 장점은 데이터의 참조가 쉽다는 점이다.
5) 스택을 이용해서 계산기 프로그램을 구현할 수 있다.
3번 문제. 다음 그림에 가장 잘 맞는 자료구조는 무엇인가? ( )



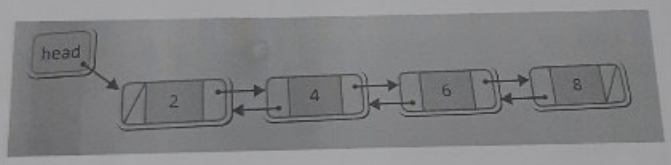
(1) 단순 연결 리스트
(2) 양방향 연결 리스트
(3) 원형 연결 리스트
(4) 순차 리스트
(5) 배열

4번 문제. <보기>에 있는 수행시간 함수들의 시간복잡도를 Big-O로 나타내고자 한다. Big-O가 작은 것부터 큰 것까지 정렬된 것을 찾아라. ( )

< 보기 >
$n^2+3n+100000,\ n!,\ 2^n, 3^n, 1, \log n$

(1) $\log n < 1 < n^2+3n+100000 < n! < 2^n < 3^n$
(2) $\log n < 1 < n^2+3n+100000 < 2^n < 3^n < n!$
(3) $1 < \log n < n^2+3n+100000 < n! < 2^n < 3^n$
(4) $1 < \log n < n^2+3n+100000 < 2^n < 3^n < n!$
(5) $n^2+3n+100000 < \log n < 1 < 2^n < 3^n < n!$

5번 문제. 다음 그림이 가장 잘 나타내는 자료구조는 무엇인가? ( )



(1) 단순 연결 리스트
(2) 양방향 연결 리스트
(3) 원형 연결 리스트
(4) 배열
(5) 순차 리스트

Input

문제 번호가 입력으로 제시된다. 1번 문제의 경우 1, 2번 문제의 경우 2가 제시된다. 문제 번호는 1부터 5 사이의 정
이다.

Output

각 문제에 대한 답을 하나의 정수로 출력한다. 예를 들어, 1번 문제의 답이 3이라면 3을 출력한다.
아래 예시는 1번 문제의 답이 3일 때와, 2번 문제의 답이 5일 때를 나타낸 것이다.
예제는 채점하지 않으며 자신이 맞다고 생각하는 답을 출력하면 된다.

Problem B

오엑스를 맞춰보자!

제한 시간 : 1초, 메모리 제한 : 128MB, 난이도 : 쉬움

여러분은 한 달간 자료구조 이론에 대해 공부했었다. 이제 다음 오엑스 문제에 대해 답할 시간이 되었다!

우리가 만들 프로그램은, 문제 번호가 입력으로 제시될 때,
그 문제에 해당하는 답을 출력하여 주는 프로그램을 완성해 보는 것이다.
이 때, 답이 O라면 1을, 답이 X라면 2를 출력하기로 하자.

현재 미완성된 프로그램은 다음과 같다. (6번 줄에서, 배열의 빈칸에 정답을 채워 넣은 소스 코드를 제출하면 된다.)

```
1   #include <iostream>
2   #define PROBLEM_NUM_MAX 5            // 최대 문제 수
3   using namespace std;
4
5   // 1번 문제의 정답은 answer[0]에, 2번 문제의 정답은 answer[1]에, ...  기입한다.
6   int answer[PROBLEM_NUM_MAX] = { ㅇ ㅇ ㅇ ㅇ ㅇ };
7
8   int main(void) {
9       int problem_number;
10      cin >> problem_number;
11      for (int i = 1; i <= PROBLEM_NUM_MAX; i++) {
12          if (problem_number == i) {
13              cout << answer[i - 1] << endl;
14              break;
15          }
16      }
17      return 0;
18  }
19
```

아래 5문제를 보고, O, X 여부를 판단하여 위 소스 코드를 완성하여라. 예를 들어, 1번 문제의 답이 O라면, 입력값이 1일 때 출력값은 1이다. 만약 3번 문제의 답이 X라면, 입력값이 3일 때 출력값은 2이다.

1번 문제. 스택은 먼저 들어간 것이 나중에 나오는 구조이다. (O)
2번 문제. 연결 리스트에서는 정렬 기능, 삽입 기능을 구현할 수 없다. (X) ?
3번 문제. 연결 리스트에서, 새 노드를 추가할 때 리스트의 머리에 저장할 경우 포인터 변수 tail이 불필요하다. (O)
4번 문제. 리스트는 연결 리스트만을 의미한다. (X)
5번 문제. O(1)은 상수형 빅-오라고 하며 데이터 수에 상관없이 연산횟수가 고정임을 의미한다. (O)

Input

문제 번호가 입력으로 제시된다. 1번 문제의 경우 1, 2번 문제의 경우 2가 제시된다. 문제 번호는 1부터 5 사이의 정수이다.

제한 시간 : 1초, 메모리 제한 : 128MB, 난이도 : 쉬움

여러분은 한 달간 자료구조 이론에 대해 공부했었다. 이제 다음 객관식 문제에 대해 답할 시간이 되었다!

우리가 만들 프로그램은, 문제 번호가 입력으로 제시될 때,
그 문제에 해당하는 답을 출력하여 주는 프로그램을 완성해 보는 것이다.

현재 미완성된 프로그램은 다음과 같다. (6번 줄에서, 배열의 빈칸에 정답을 채워 넣은 소스 코드를 제출하면 된다.)

```
1   #include <iostream>
2   #define PROBLEM_NUM_MAX 5            // 최대 문제 수
3   using namespace std;
4
5   // 1번 문제의 정답은 answer[0]에, 2번 문제의 정답은 answer[1]에, ...  기입한다.
6   int answer[PROBLEM_NUM_MAX] = { ㅇ ㅇ ㅇ ㅇ ㅇ };
7
8   int main(void) {
9       int problem_number;
10      cin >> problem_number;
11      for (int i = 1; i <= PROBLEM_NUM_MAX; i++) {
12          if (problem_number == i) {
13              cout << answer[i - 1] << endl;
14              break;
15          }
16      }
17      return 0;
18  }
19
```

아래 5문제를 보고, 위 소스 코드를 완성하여라.

1번 문제. 다음 중 틀린 것은? (2)

(1) 이진 탐색 알고리즘(binary search algorithm)보다 순차 탐색 알고리즘(sequential search algorithm)이 더 느리다.
(2) ADT(Abstract Data Type)는 추상 자료형이라고 하며, 구체적인 기능의 완성과정을 언급한 것이다.
(3) 리스트는 배열을 통해서도 구현할 수 있다.
(4) 연결 리스트에서 더미 노드(Dummy Node)를 사용하는 이유는 노드의 추가, 삭제 및 조회 과정을 일관되게 구성할 수 있기 때문이다.
(5) 큐는 선입선출(FIFO)의 성질을 지닌다. 즉, 먼저 들어온 값이 먼저 나가게 된다.

2번 문제. 다음 보기 중 가장 적절하지 않은 것은? (4)

(1) 재귀함수의 문제점 중 하나는 중복된 계산이 시행될 수 있다는 점이다.
(2) T(n) = 3n + 2일 때, O(n) = n이다.
(3) 일반적인 이진 탐색 알고리즘의 시간 복잡도는 O(log n)이다.
(4) 리스트를 자료형과 포인터 변수를 이용하여 구현하는 것의 장점은 데이터의 참조가 쉽다는 점이다.
(5) 스택을 이용해서 계산기 프로그램을 구현할 수 있다.

4번 문제. <보기>에 있는 수행시간 함수들의 시간복잡도를 Big-O로 나타내고자 한다. Big-O가 작은 것부터 큰 것까지 정렬된 것을 찾아라. ( )

< 보기 >

$n^2+3n+100000,\ n!,\ 2^n,\ 3^n,\ 1,\ \log n$

(1) $\log n < 1 < n^2+3n+100000 < n! < 2^n < 3^n$

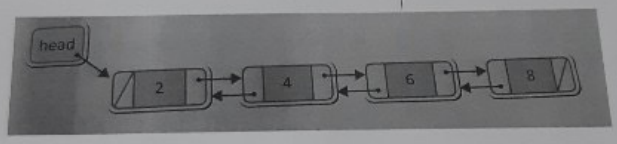
(2) $\log n < 1 < n^2+3n+100000 < 2^n < 3^n < n!$

(3) $1 < \log n < n^2+3n+100000 < n! < 2^n < 3^n$

(4) $1 < \log n < n^2+3n+100000 < 2^n < 3^n < n!$

(5) $n^2+3n+100000 < \log n < 1 < 2^n < 3^n < n!$

5번 문제. 다음 그림이 가장 잘 나타내는 자료구조는 무엇인가? ( )



(1) 단순 연결 리스트
(2) 양방향 연결 리스트
(3) 원형 연결 리스트
(4) 배열
(5) 순차 리스트

Input

문제 번호가 입력으로 제시된다. 1번 문제의 경우 1, 2번 문제의 경우 2가 제시된다. 문제 번호는 1부터 5 사이의 이다.

Output

각 문제에 대한 답을 하나의 정수로 출력한다. 예를 들어, 1번 문제의 답이 3이라면 3을 출력한다.
아래 예시는 1번 문제의 답이 3일 때와, 2번 문제의 답이 5일 때를 나타낸 것이다.
예제는 채점하지 않으며 자신이 맞다고 생각하는 답을 출력하면 된다.