20191631 윤민상 퀴즈 만점 문제풀이
<QUIZ1 문제>

QUIZ #1 (40점 만점)

1. 시스템 생명주기 6단계를 설명하시오(5점)

2. 알고리즘의 5가지 조건을 설명하시오(5점)

3. 이진탐색 알고리즘을 **반복문을 사용한 코드와 재귀형식의 코드로** 각각 작성하시오 (10점)

4. 다음 fibonacci 알고리즘의 **1)step count 및 big-O 수행시간을** 구하고, **2)recursive 알고리즘을** 작성하라 (10점)

```
void fibonacci(int n)
{  int i, fibo, fibo1, fibo2;
    if(n <= 1)
          printf("%d", n);
   else {
      fibo1 = 1;  fibo2 = 0;
      for (i = 2; i <= n; i++) {
         fibo = fibo1 + fibo2;
         fibo2 = fibo1;
                fibo1 = fibo;
      }
       printf("%d", fibo);
   } }
```

5. Magic Square 는 1에서 n2까지의 정수로된 nxn행렬로서, 각행의 합, 열의 합, 주 대각선의 합이 모두 같다. 크기가 5인(5x5) Magic Square를 작성하시오 (**조건: 첫번째 행의 중앙에 1을 넣는다. 왼쪽위로 이동함**. 알고리즘 작성은 필요 없음, 표를 완성할 것) (10점)

<QUIZ1 답>

1. 요구사항 분석 -> 명세 -> 설계 -> 구현 -> 검증 -> 운영 및 유지보수

요구사항 분석 : 문제에 대한 적절한 해를 구하기 위한 요구조건을 정의

명세 : 시스템이 무엇을 해야하는가를 정의

설계 : 명세된 기능을 어떻게 수행하는가 기술

구현 : 코딩

검증 : 프로그램의 수행, 성능, 정확성 검증

운영 및 유지보수 : 시스템 설치, 운영, 유지보수

2.

입력 : 0 또는 더 큰 값이 외부 제공

출력 : 적어도 한 개 이상의 결과 생성

명확성 : 모호하지 않은 명확한 명령

유한성 : 종료

유효성 : 기본적, 실행가능 명령

3.

반복문을 사용한 코드 :

```
int iterativeBinarySearch(int* list, int count, int num, int left, int right)
{
   while (left <= right)
   {
      int middle = (left + right) / 2;
      if (num < list[middle])
         right = middle - 1;
      else if (num == list[middle])
         return middle;
      else
         left = middle + 1;
   }
   return -1;
}
```

재귀형식의 코드 :

```
int recursiveBinarySearch(int* list, int count, int num, int left,int right)
{
   while (left <= right)
   {
      int middle = (left + right) / 2;
      switch (compare(list[middle], num))
      {
      case -1:
         left = middle + 1;
         break;
      case 0:
         return middle;
      case 1:
         right = middle - 1;
         break;
      }
   }
   return -1;
}
```

참고 : compare 함수는 따로 생성!

```
int compare(int x, int y)
{
   if (x > y) return 1;
   else if (x < y) return -1;
   else return 0;
}
```

4.

1) step count : 4n-2 / big-O 수행시간 : O(n)

2) recursive 알고리즘

```
int fibonacci(int n) {
    if (n == 1){
       return 1;}

   else if (n == 0) {
      return 0;}

   else {
      return fibo(n - 1) + fibo(n - 2);}
```

5.

15 8 1 24 17

16 14 7 5 23

22 20 13 6 4

3 21 19 12 10

9 2 25 18 11