1. 완전수는 해당 숫자에 대한 약수의 합이 자기 자신이 되는 경우를 말한다. 즉, 자연수 6은 1, 2, 3이 그 약수이며, 아래와 같은 경우이므로 완전수라 할 수 있다.

   6 = 1 + 2 + 3

   이에 비해 자연수 10은 아래와 같이 해당 숫자에 대한 약수의 합이 자기 자신이 되지 않기 때문에 완전수라 할 수 없다.

   $10 \neq 1 + 2 + 5$

   이와 같은 원리에 따라 0부터 입력받은 숫자 내에서 완전수를 구하여 출력하는 코드를 완성하라.

   즉, 사용자에게 10000을 입력 받았다면, 0부터 10,000까지의 숫자 중 완전수를 작은 수부터 출력하면 된다.

실행 결과 예

```
Please input the number : 10
The result is  [6]
Please input the number : 50
The result is  [6, 28]
Please input the number : 500
The result is  [6, 28, 496]
```

2. 아래는 버블정렬를 설명하기 위해 순서를 나타낸 것이다. 비어있는 코드를 작성하여, 버블정렬을 완성하도록 하자 [6점].

참고: 버블정렬

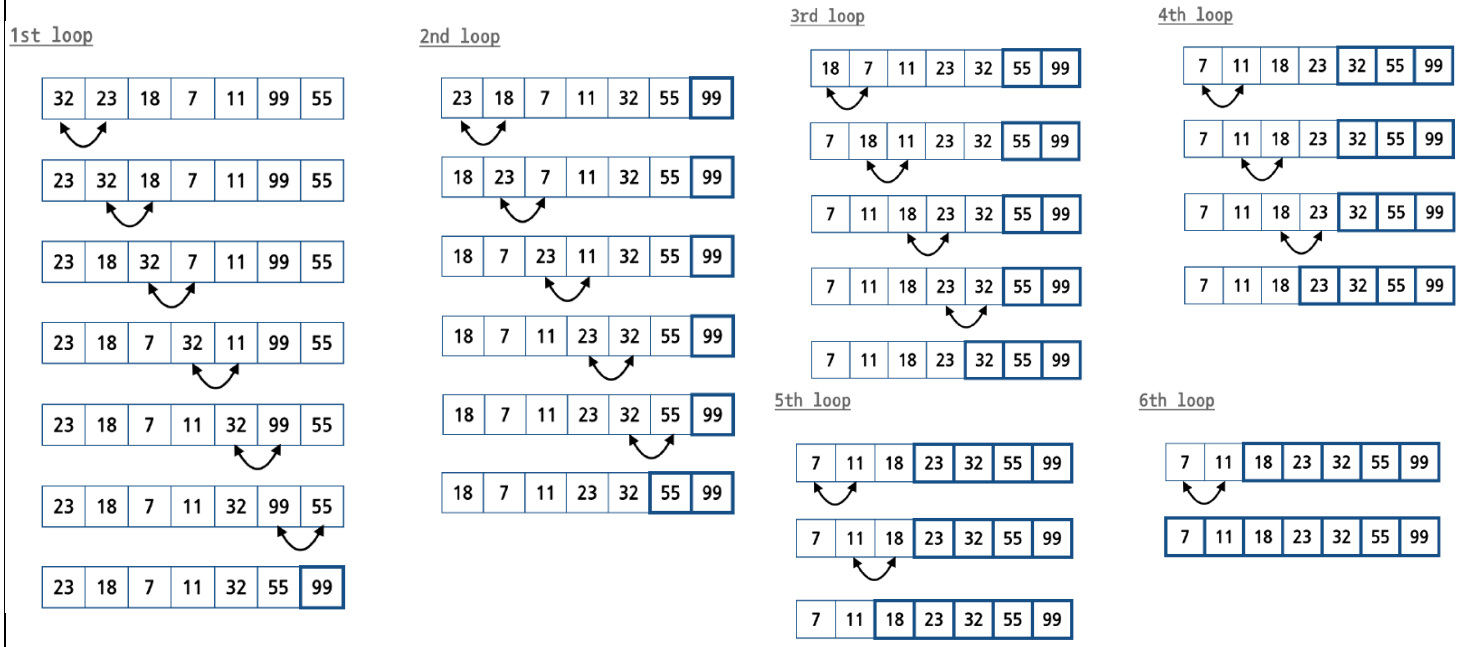
- 리스트에서 인접한 두 수를 비교하여 순서가 바뀐 경우 이를 교환 정렬함
- 제자리정렬(in-place sort)이라고도 불림
  > 추가공간을 사용하지 않고 자체적으로 값을 교환 정렬
  > 버블정렬은 리스트 내부에서 수의 교환이 필요해서 추가 공간이 필요하지 않음

실행 결과 예

[45, 64, 36, 13, 55, 63, 67, 33, 47, 61]

[13, 33, 36, 45, 47, 55, 61, 63, 64, 67]

순서 예시



아래는 버블정렬 Test 코드 입니다. 빈칸을 채워넣으면 됩니다. 아래 테스트 코드는 있는 그대로 입력하시면 됩니다.

```
def bubble_sort(originlist):

(빈 칸)



#Do not revise or touch!
import random
originlist = random.sample(range(100), 10)
print(originlist)
afterlist = bubble_sort(originlist)
print(afterlist)
```

3. 1에서부터 n까지의 곱을 나타내는 n! (n factorial)을 계산하는 함수를 작성하고자 한다.
   3-1) 재귀함수를 활용하여 n!를 계산하는 fac1 함수를 완성시켜 보자
   3-2) 이 구조를 활용하여 for 반복문을 사용한 fac2 함수를 완성시켜 보자
   3-3) 이 구조를 활용하여 while 반복문을 사용한 fac3 함수를 완성시켜 보자
   - fac0는 무시할 것 (결과 값을 위한 샘플)

실행 결과 예

```
Please input 'n' for the factorial : 5
The result of fac0 is  120
The result of fac1 is  120
The result of fac2 is  120
The result of fac3 is  120
```

뼈대 코드는 다음과 같다.

==

```
def fac1(n):

def fac2(n):

def fac3(n):

n = int(input("Please input 'n' for the factorial : "))

a = fac1(n)
print("The result of fac1 is ",a)

a = fac2(n)
print("The result of fac2 is ",a)

a = fac3(n)
print("The result of fac3 is ",a)
```

4. 특정 수업의 최종 성적 (100점 기준)을 저장한 리스트 grade_result 가 있다고 하자. 이 리스트 내에는 해당 수업을 수강한 학생 20명의 최종 성적이 저장되어 있다. 이 성적을 기반으로 아래와 같은 기준으로 학점을 부여하고 평균평점, 성취도를 계산하고자 한다.

   함수 student_grading: 95점 이상은 A+, 90점 이상은 A, 85점 이상은 B+, 80점 이상은 B, 75점 이상은 C+, 나머지는 C를 부여한다. 그리고 이를 기반으로 각 학점 별 학생 숫자를 출력하자. 그 후 부여 학점을 기준으로 해당 수업의 평균평점을 구하자. A+는 4.5, A는 4.0, B+는 3.5, B는 3.0, C+는 2.5, C는 2.0으로 환산한 후, 해당 수업의 평균평점을 구하고 출력하자 [12점].

실행 결과 예

```
A+ : 3
A : 2
B+ : 2
B : 3
C+ : 4
C : 6
The class average is  2.975
```

```
A+ : 5
A : 4
B+ : 2
B : 3
C+ : 3
C : 3
The class average is  3.4
```

뼈대 코드는 다음과 같다. 빈칸을 채우면 된다.

```
def student_grading(grade_result):
 (빈 칸)

#Do not touch!
import random
grade_result = []
grade_result = random.sample(range(70,101), 20)
student_grading(grade_result)
```