1. 1 부터 100 사이의 홀수를 만들어 변수 x 에 저장하고 출력하시오.
2. x 를 5 행 10 열의 배열로 변환하여 y 에 저장하고 출력하시오.
3. y 의 차원을 확인하시오.
4. y 의 배열의 모양(shape)을 확인하시오.
5. y 의 크기(총 몇 개의 값으로 구성되었는지)를 구하시오.
6. x 에서 마지막 10 개의 값을 출력하시오
7. x 의 값을 하나 걸러 하나씩 출력하시오
8. x 의 값을 뒤에서부터 하나 걸러 하나씩 출력하시오
9. 7 과 8 의 결과를 곱하여 z 에 저장하시오
10. 2,4,6,8,10 의 숫자 패턴을 5 번 반복하는 배열을 만들어 w 에 저장하고 출력하시오.
11. x 와 w 를 결합하여 새로운 배열 k 를 만드시오.
12. k 에 있는 모든 요소의 합을 구하고 결과를 출력하시오
13. k 의 평균을 구하고 결과를 출력하시오
14. k 의 중앙값을 구하고 결과를 출력하시오
15. k 의 표준편차를 구하고 결과를 출력하시오
16. k 안에 있는 값 중 최대값과 최소값을 구하고 출력하시오.
17. k 의 값 중 평균보다 큰 값의 개수를 구하시오
18. k 의 값 중 평균보다 큰 값만 출력하시오
19. k 의 값 중 평균 보다 크고 평균에 표준편차를 더한 값보다 작은 값만 출력하시오.
20. k 를 오름차순으로 정렬하고 결과를 출력하시오.
21. k 를 내림차순으로 정렬하고 결과를 출력하시오.
22. 0 으로 채워진 3 행 2 열의 배열을 만들고 출력하시오.
23. 1 로 채워진 2 행 3 열의 배열을 만들고 출력하시오.
24. 5 로 채워진 길이가 10 인 1 차원 배열을 만들고 출력하시오.
25. 2,4,6,8,10 을 각 숫자 요소별 5 번씩 반복하는 배열을 만들고 출력하시오.
26. 2,4,6,8,10 의 숫자 패턴을 5 번 반복하는 배열을 만들고 출력하시오.

[1, 2, 3, 4, 5]

```
# 1. ---> 1부터 100사이의 홀수를 만들어 변수 x에 저장하고 출력하시오.
print('# 1')
x = np.arange(1, 100, 2)
print(x)
print()

# 2. ---> x를 5행 10열의 배열로 변환하여 y에 저장하고 출력하시오.
print('# 2')
y = x.reshape(5, 10)
print(y)
print()

# 3. ---> y의 차원을 확인하시오.
print('# 3')
print(np.ndim(y))
print()

# 4. ---> y의 배열의 모양(shape)을 확인하시오.
print('# 4')
print(np.shape(y))
print()

# 5. ---> y의 크기(총 몇 개의 값으로 구성되었는지)를 구하시오.
print('# 5')
print(np.size(y))
print()
```

```
# 1
[ 1  3  5  7  9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47
 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95
 97 99]

# 2
[[ 1  3  5  7  9 11 13 15 17 19]
 [21 23 25 27 29 31 33 35 37 39]
 [41 43 45 47 49 51 53 55 57 59]
 [61 63 65 67 69 71 73 75 77 79]
 [81 83 85 87 89 91 93 95 97 99]]

# 3
2

# 4
(5, 10)

# 5
50
```

[6, 7, 8, 9, 10]

```
# 6. ---> x에서 마지막 10개의 값을 출력하시오
print('# 6')
print(x[len(x) - 10:])
print()

# 7. ---> x의 값을 하나 걸러 하나씩 출력하시오
print('# 7')
print(x[::2])
print()

# 8. ---> x의 값을 뒤에서부터 하나 걸러 하나씩 출력하시오
print('# 8')
print(x[::-2])
print()

# 9. ---> 7과 8의 결과를 곱하여 z에 저장하시오
print('# 9')
z = x[::2] * x[::-2]
print(z)
print()

# 10. ---> 2,4,6,8,10 의 숫자 패턴을 5번 반복하는 배열을 만들어 w에 저장하고 출력하시오.
print('# 10')
tmp = np.arange(2, 11, 2)
w = np.tile(tmp, 5)
print(w)
print()
```

```
# 6
[81 83 85 87 89 91 93 95 97 99]

# 7
[ 1  5  9 13 17 21 25 29 33 37 41 45 49 53 57 61 65 69 73 77 81 85 89 93
 97]

# 8
[99 95 91 87 83 79 75 71 67 63 59 55 51 47 43 39 35 31 27 23 19 15 11  7
  3]

# 9
[  99  475  819 1131 1411 1659 1875 2059 2211 2331 2419 2475 2499 2491
 2451 2379 2275 2139 1971 1771 1539 1275  979  651  291]

# 10
[ 2  4  6  8 10  2  4  6  8 10  2  4  6  8 10  2  4  6  8 10  2  4  6  8
 10]
```

[11, 12, 13, 14, 15]

```
# 11. ⟶ x와 w를 결합하여 새로운 배열 k를 만드시오.
print('# 11')
k = np.concatenate([x, w])
print(k)
print()

# 12. ⟶ k에 있는 모든 요소의 합을 구하고 결과를 출력하시오
print('# 12')
add = np.sum(k)
print(add)
print()

# 13. ⟶ k 의 평균을 구하고 결과를 출력하시오
print('# 13')
avg = np.mean(k)
print(avg)
print()

# 14. ⟶ k의 중앙값을 구하고 결과를 출력하시오
print('# 14')
mid = np.median(k)
print(mid)
print()

# 15. ⟶ k의 표준편차를 구하고 결과를 출력하시오
print('# 15')
std = np.std(k)
print(std)
print()
```

```
# 11
[ 1  3  5  7  9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47
 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95
 97 99  2  4  6  8 10  2  4  6  8 10  2  4  6  8 10  2  4  6  8 10  2  4
  6  8 10]

# 12
2650

# 13
35.333333333333336

# 14
25.0

# 15
31.436002007606216
```

[16, 17, 18, 19, 20]

```
# 16.→k 안에 있는 값 중 최대값과 최소값을 구하고 출력하시오.
print('# 16')
max_v = np.max(k)
min_v = np.min(k)
print(max_v)
print(min_v)
print()

# 17.→k의 값 중 평균보다 큰 값의 개수를 구하시오
print('# 17')
print(len(k[k > avg]))
print()

# 18.→k의 값 중 평균보다 큰 값만 출력하시오
print('# 18')
print(k[k > avg])
print()

# 19.→k의 값 중 평균 보다 크고 평균에 표준편차를 더한 값보다 작은 값만 출력하시오.
print('# 19')
print(k[(k > avg) & (k < avg + std)])
print()

# 20.→k를 오름차순으로 정렬하고 결과를 출력하시오.
print('# 20')
sorted_k = np.sort(k)
print(sorted_k)
print()
```

```
# 16
99
1

# 17
32

# 18
[37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83
 85 87 89 91 93 95 97 99]

# 19
[37 39 41 43 45 47 49 51 53 55 57 59 61 63 65]

# 20
[ 1  2  2  2  2  2  3  4  4  4  4  4  5  6  6  6  6  6  7  8  8  8  8  8
  9 10 10 10 10 10 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45
 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93
 95 97 99]
```

[21, 22, 23, 24, 25, 26]

```
# 21. ---> k를 내림차순으로 정렬하고 결과를 출력하시오.
print('# 21')
r_sorted_k = sorted_k[::-1]
print(r_sorted_k)
print()

# 22. ---> 0으로 채워진 3행 2열의 배열을 만들고 출력하시오.
print('# 22')
fill_zero = np.zeros(6).reshape(3, 2)
print(fill_zero)
print()

# 23. ---> 1로 채워진 2행 3열의 배열을 만들고 출력하시오.
print('# 23')
fill_one = np.ones(6).reshape(2, 3)
print(fill_one)
print()

# 24. ---> 5로 채워진 길이가 10인 1차원 배열을 만들고 출력하시오.
print('# 24')
fill_five = np.full(10, 5)
print(fill_five)
print()

# 25. ---> 2,4,6,8,10 을 각 숫자 요소별 5번씩 반복하는 배열을 만들고 출력하시오.
print('# 25')
x = np.arange(2, 11, 2)
np_repeat = np.repeat(x, 5)
print(np_repeat)
print()

# 26. ---> 2,4,6,8,10 의 숫자 패턴을 5번 반복하는 배열을 만들고 출력하시오.
print('# 26')
x = np.arange(2, 11, 2)
np_tile = np.tile(x, 5)
print(np_tile)
```

```
# 21
[99 97 95 93 91 89 87 85 83 81 79 77 75 73 71 69 67 65 63 61 59 57 55 53
 51 49 47 45 43 41 39 37 35 33 31 29 27 25 23 21 19 17 15 13 11 10 10 10
 10 10  9  8  8  8  8  8  7  6  6  6  6  6  5  4  4  4  4  4  3  2  2  2
  2  2  1]

# 22
[[0. 0.]
 [0. 0.]
 [0. 0.]]

# 23
[[1. 1. 1.]
 [1. 1. 1.]]

# 24
[5 5 5 5 5 5 5 5 5 5]

# 25
[ 2  2  2  2  2  4  4  4  4  4  6  6  6  6  6  8  8  8  8  8 10 10 10 10
 10]

# 26
[ 2  4  6  8 10  2  4  6  8 10  2  4  6  8 10  2  4  6  8 10  2  4  6  8
 10]
```