Run Length Encoding

CODE:

```
def algo(arr):
      count = 1
      prev = arr[0]
      ans = []
      for ele in arr[1:]:
      if ele == prev:
            count += 1
      else:
            ans.extend((prev, count))
            count  = 1
      prev = ele
      else:
      ans.extend((prev, count))
      return ans


if __name__ == '__main__':
      ip = tuple(map(int, input('enter the ip sequence(without spaces):
')))
      op = algo(ip)

      print('output sequence:', op)
      print('original space requirement:', len(ip) * 8)
      print('space requirement of output:', len(op) * 8)
```

OUTPUT:

CASE 1: When RLE actually reduces the space requirement.

```
enter the ip sequence(without spaces):
11111111100000101000001011111111000000111
output sequence: [1, 9, 0, 5, 1, 1, 0, 1, 1, 1, 0, 5, 1, 1, 0, 1, 1, 8,
0, 6, 1, 3]
original space requirement: 328
space requirement of output: 176
```

CASE 1: When space required by RLE o/p requires more space than uncompressed input.

```
enter the ip sequence(without spaces):
10101010101010101001001010100101111110000
output sequence: [1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1,
0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 2, 1, 1, 0, 2, 1, 1,
0, 1, 1, 1, 0, 1, 1, 1, 0, 2, 1, 1, 0, 1, 1, 5, 0, 4]
original space requirement: 320
```

```
space requirement of output: 480
```