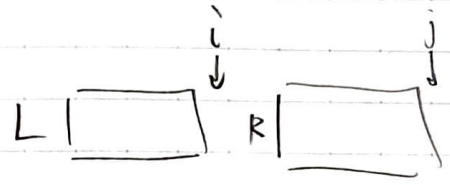
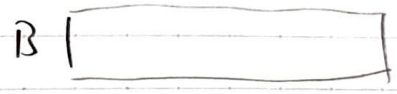
2019 アルゴ

1 (1)
```
func(A){
  n = len(A)
  if (n==1)  return A

  m = floor(n/2)          n=偶数
  L = func(A[0:m])        = m
  R = func(A[m:n])        = n-1-(m-1) = n-m
                          i=偶 j=奇
  B = []   i,j = 0 0.
  while(i+j < n
    if (i==len(L) and j<len(R))
      B.append(R[j])  j++
    elif (j==len(R) and i<len(L))
      B.append(L[i])  i++
    elif(L[i] ≦ R[j])
      B.append(L[i])  i++
    else
      B.append(R[j])  j++
  print(B)
  return B
```





マージソート

(2) ①[2|9|5|3|7|0|1|4]

[2, 9]
[3, 5]
[2, 3, 5, 9]
[0, 7]
[1, 4]
[0, 1, 4, 7]
[0, 1, 2, 3, 4, 5, 7, 9]

(3) 求める時間計算量を $T(n)$ とすると

$$T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{2}\right) + O(n)$$

が成り立つ。ここで $T(1) = O(1)$, $n = 2^k$ とする。

$$T(n) = 2T\left(\frac{n}{2}\right) + O(n)$$
$$= 2\left(2T\left(\frac{n}{2^2}\right) + O\left(\frac{n}{2}\right)\right) + O(n)$$
$$= 2\left(2\left(\cdots\left(2T\left(\frac{n}{2^k}\right) + O\left(\frac{2^k n}{2^k}\right)\right) + \cdots\right) + O(n)$$
$$= 2^k T\left(\frac{n}{2^k}\right) + 2^{k-1} O\left(\frac{n}{2^{k-1}}\right) + \cdots + 2O\left(\frac{n}{2}\right) + O(n)$$
$$= n + 2^{k-1} C\frac{n}{2^{k-1}} + \cdots + 2C\frac{n}{2} + Cn$$
$$= n + Cn(1 \times k)$$
$$= n + Cn\log_2 n$$
$$= O(n \log n)$$

$$2\left(2\left(2T\left(\frac{n}{2^3}\right) + O\left(\frac{n}{2^2}\right)\right) + O\left(\frac{n}{2}\right)\right) + O(n)$$
$$= 2^3 T\left(\frac{n}{2^3}\right) + 2^2 O\left(\frac{n}{2^2}\right) + 2O\left(\frac{n}{2}\right) + O(n)$$

KMP

2 (1) (i)

j	1	2	3	4
shift	1	2	3	1

(ii)

j	1	2	3	4
shift	1	1	~~2~~ 3	2

(2) text: ABABBAABACA
pattern: ABAC
ABAC
~~ABAC~~
ABAC
ABAC 一致

(3) j=mまで比較して不一致で shift[j]=1のとき最悪
よってこのとき比較回数は O(m×n)

また比較の最大回数は m×(n−m+1)回
その例は text: AAAAAAAAA
pattern: AAAB
解説
AAAB 4
A
AAAB 4×(8−4+1)=20回

C□ AC□
AAAB AAAB t[i+1]≠p[2]=p[1]より
AA 1つとばせる

AAC□ AAAAB 次もかも
AAAB AAAB
A AAA

B□ AA ABC ABAB
ABAC ABAC ABAC ABAC
AB × A
AB

AAAAAAAAA
AAAB
AAAB
AAAB AAAB

AAAAAAAAAA
ABCD