Answer to Question No.1

StudentId = 20101238
a= 38
b= 20
BirthYear = 2000

A.

Here, start_index = 6

Value	26	25	53	52	0	0	45	45	25	5	19	43	5
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	26	25	53	52	0	0	0	45	25	5	19	43	5
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	5	26	53	52	0	0	0	45	45	25	5	19	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	5	26	53	52	0	0	0	0	45	45	5	19	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	5	26	53	52	0	0	0	0	0	45	5	19	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

B.

Value	5	26	53	52	0	0	0	0	0	45	5	19	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇
Start index

Here the start index of this array is 9.
b%67= 20%67=20
Position 5 means index=1.

Value	5	26	53	52	52	0	0	0	0	45	5	19	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	5	26	53	53	52	0	0	0	0	45	5	19	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	5	26	26	53	52	0	0	0	0	45	5	19	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	5	20% 67= 20	26	53	52	0	0	0	0	45	5	19	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	5	20	26	53	52	0	0	0	0	45	5	19	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

C.
Start index = 9
StudentId%13=20101238%13=1
Position 8 means index = 4

Value	5	20	26	53	52	52	0	0	0	45	5	19	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

Value	5	20	26	53	1	52	0	0	0	45	5	19	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

D.
Start index = 9
BirthYear = 2000
BirthYear%61=2000%61=48
Position 3 means index=12

Value	5	20	26	53	1	52	52	0	0	45	5	19	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

Value	5	20	26	53	1	1	52	0	0	45	5	19	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

Value	5	20	26	53	53	1	52	0	0	45	5	19	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

Value	5	20	26	26	53	1	52	0	0	45	5	19	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

Value	5	20	20	26	53	1	52	0	0	45	5	19	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	5	5	20	26	53	1	52	0	0	45	5	19	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	43	5	20	26	53	1	52	0	0	45	5	19	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	43	5	20	26	53	1	52	0	0	45	5	19	48
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

E.

Value	43	5	20	26	53	1	52	0	0	45	5	19	48
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	43	5	20	26	53	1	52	0	0	45	19	19	48
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	43	5	20	26	53	1	52	0	0	45	19	48	48
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Similarly,

Value	5	20	26	53	1	52	52	0	0	45	19	48	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	5	20	26	53	1	52	0	0	0	45	19	48	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	20	20	26	53	1	52	0	0	0	45	19	48	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	20	26	26	53	1	52	0	0	0	45	19	48	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	20	26	53	1	52	52	0	0	0	45	19	48	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	20	26	53	1	52	0	0	0	0	45	19	48	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

F.

Value	20	26	53	1	0	0	0	0	0	45	19	48	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

G.

Value	20	26	53	1	0	0	0	0	0	45	19	48	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	20	26	53	1	1	0	0	0	0	45	19	48	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	20	26	53	53	1	0	0	0	0	45	19	48	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	20	26	26	53	1	0	0	0	0	45	19	48	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Similarly,

Value	43	20	26	53	1	0	0	0	0	45	45	19	48
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Index 9 value = 0

Value	43	20	26	53	1	0	0	0	0	0	45	19	48
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

Time = 2

Value	43	20	26	53	1	1	0	0	0	0	45	19	48
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	43	20	26	53	53	1	0	0	0	0	45	19	48
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Similarly,

Value	48	43	20	26	53	1	0	0	0	0	45	45	19

Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Setting index 10 value = 0

Value	48	43	20	26	53	1	0	0	0	0	0	45	19
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

time=3

Value	48	43	20	26	53	1	0	0	0	0	0	45	19
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	48	43	20	26	53	1	1	0	0	0	0	45	19
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	48	43	20	26	53	53	1	0	0	0	0	45	19
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Similarly,

Value	19	48	43	20	26	53	1	0	0	0	0	45	45
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Setting index 11 value = 0

Value	19	48	43	20	26	53	1	0	0	0	0	0	45
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

H.

Time =1

Value	19	48	43	20	26	53	1	0	0	0	0	0	45
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	19	48	43	20	26	53	1	0	0	0	0	45	45
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	19	48	43	20	26	53	1	0	0	0	0	45	19
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Similarly,

Value	48	43	20	26	53	1	1	0	0	0	0	45	19
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Setting index 6 value = 0

Value	48	43	20	26	53	1	0	0	0	0	0	45	19
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

time=2

Value	48	43	20	26	53	1	0	0	0	0	45	45	19
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	48	43	20	26	53	1	0	0	0	0	45	19	19
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Similarly,

Value	43	20	26	53	1	1	0	0	0	0	45	19	48
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Setting index 5 value = 0,

Value	43	20	26	53	1	0	0	0	0	0	45	19	48
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

time=3

Value	43	20	26	53	1	0	0	0	0	45	45	19	48
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	43	20	26	53	1	0	0	0	0	45	19	19	48
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Similarly,

Value	20	26	53	1	1	0	0	0	0	45	19	48	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Setting index 4 value = 0,

Value	20	26	53	1	0	0	0	0	0	45	19	48	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

time = 4

Value	20	26	53	1	0	0	0	0	45	45	19	48	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Value	20	26	53	1	0	0	0	0	45	19	19	48	43
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Similarly,

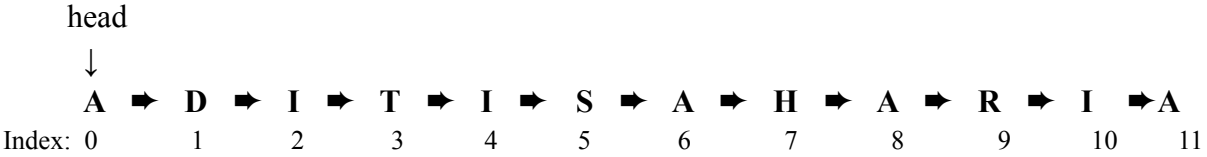
Value	26	53	1	1	0	0	0	0	45	19	48	43	20
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

⬇

Setting index 3 value = 0,

Value	26	53	1	0	0	0	0	0	45	19	48	43	20
Index	0	1	2	3	4	5	6	7	8	9	10	11	12

Answer to Question No.3

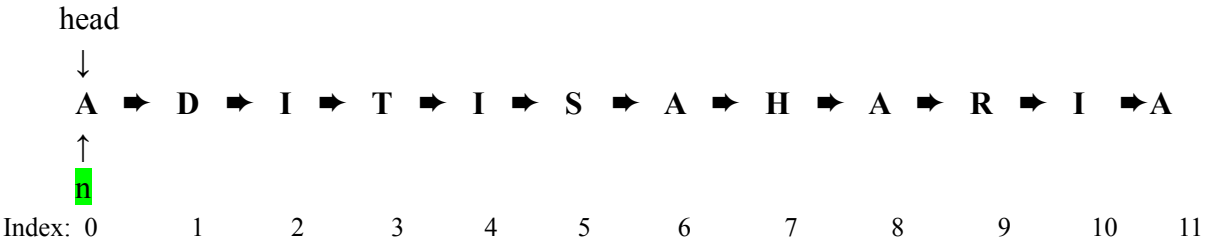


A.

new = None

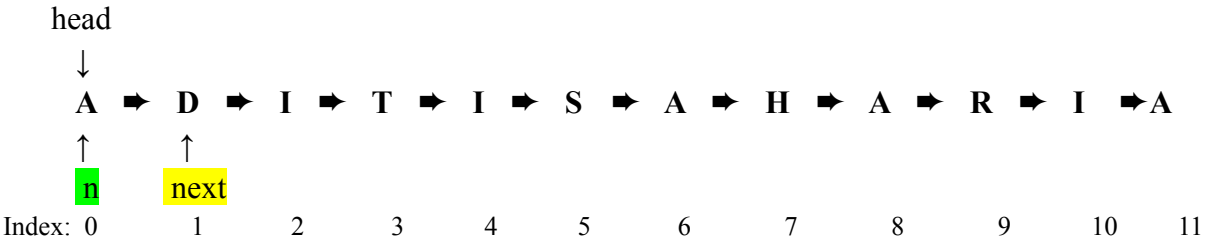
Step-01:

n=head

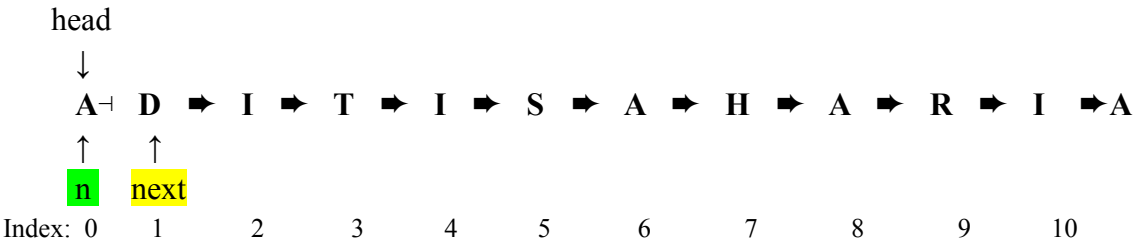


Step-02:

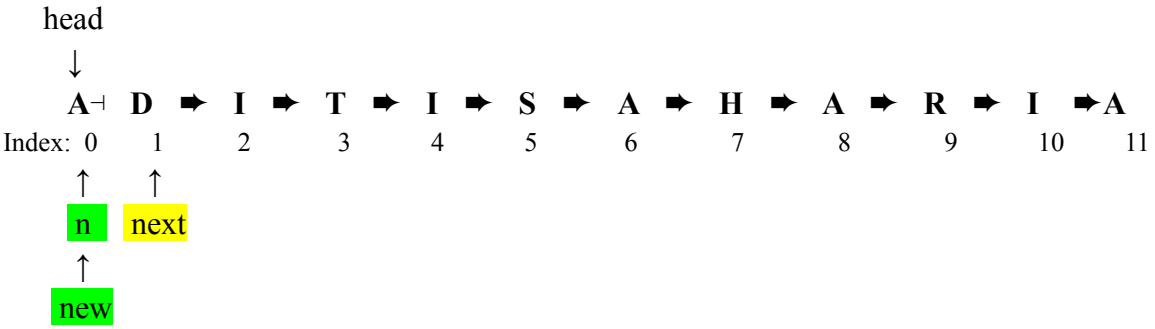
next = n.next



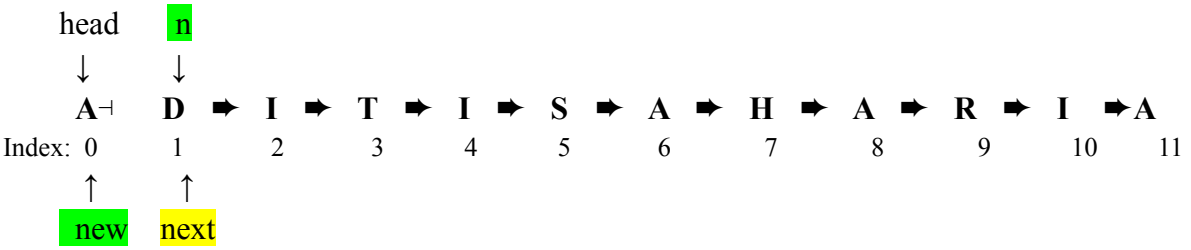
n.next = new



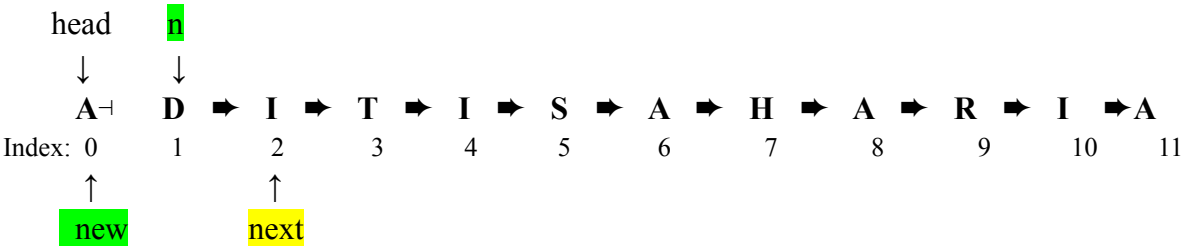
new = n



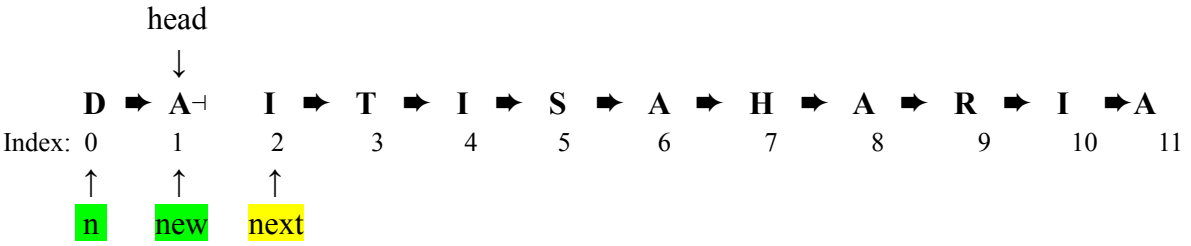
n = next



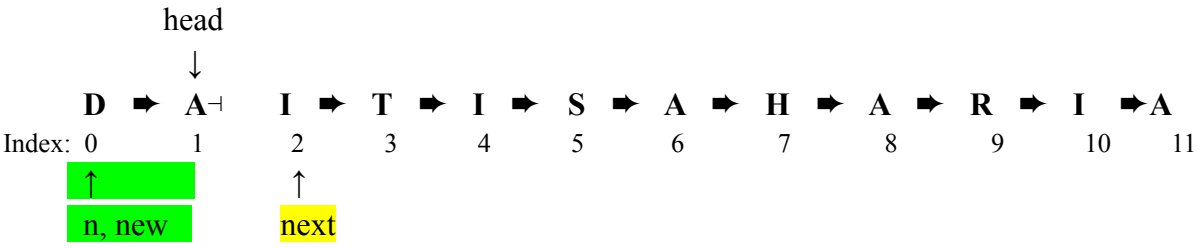
Again repeating the steps,



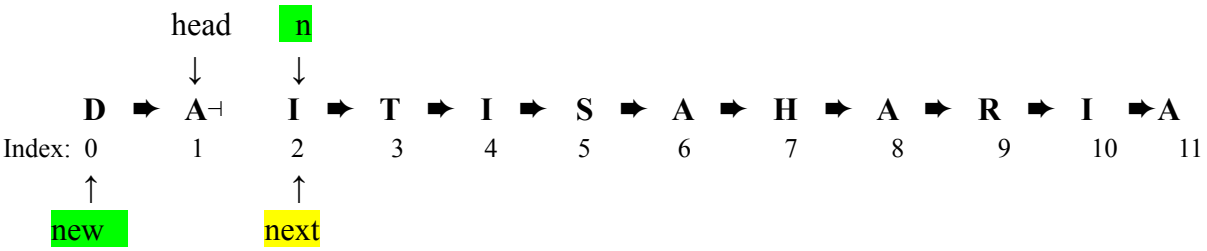
As, n.next = new so,



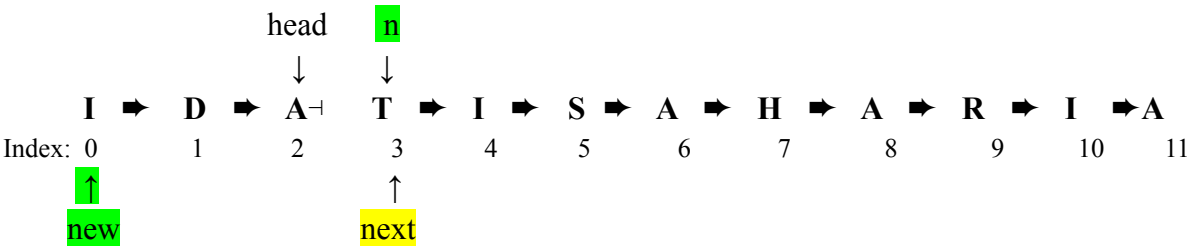
As, new = n so,



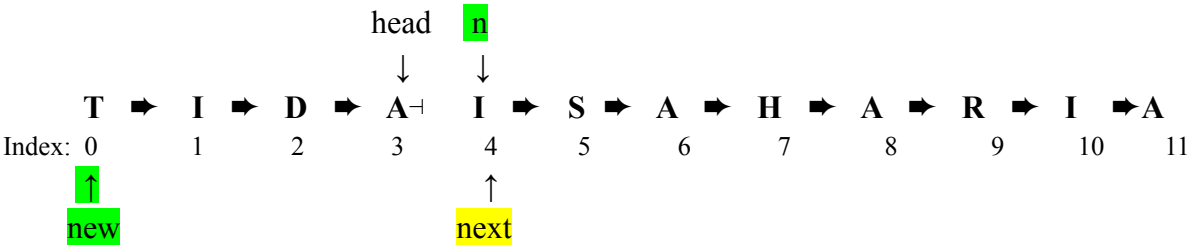
As n=next so,



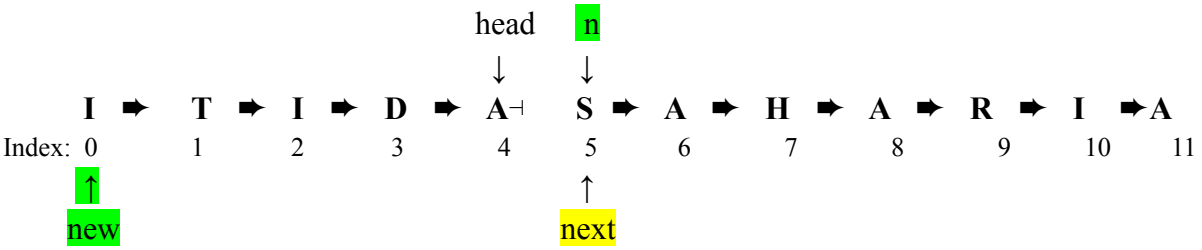
Again,



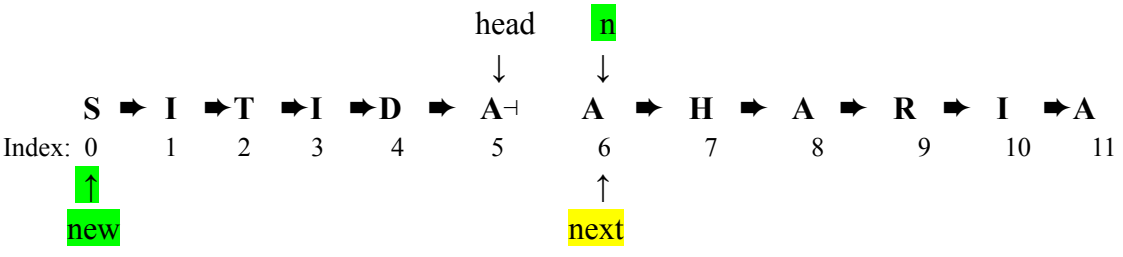
Again,



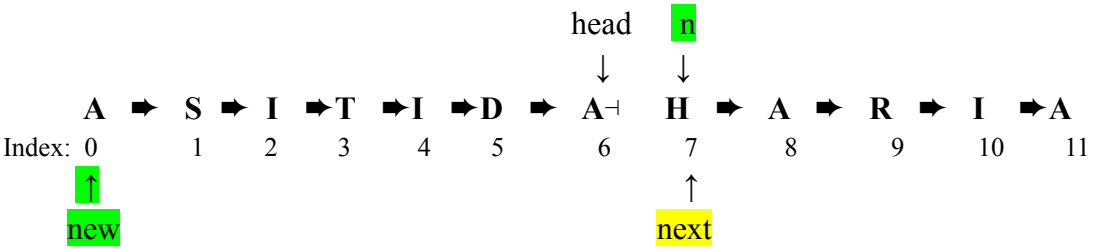
Again,



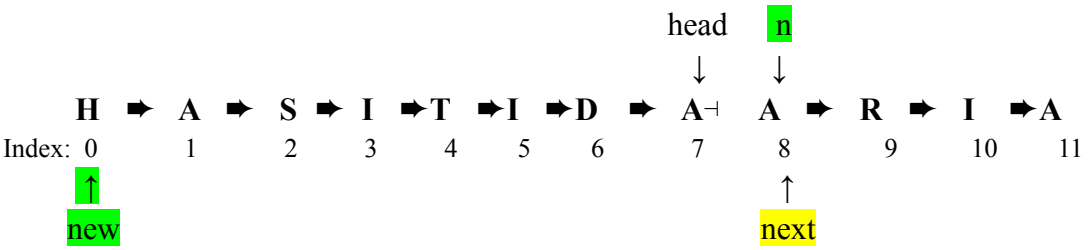
Again,



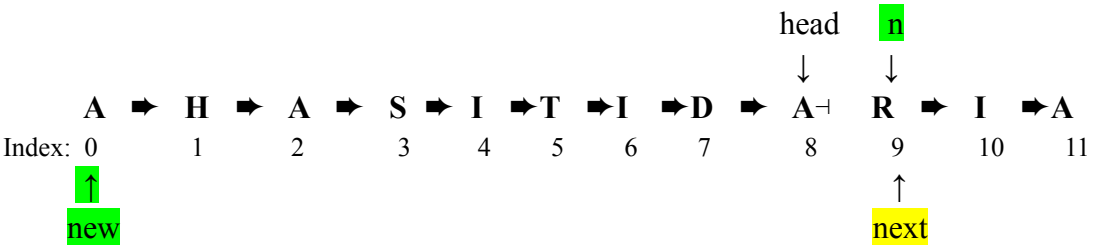
Again,



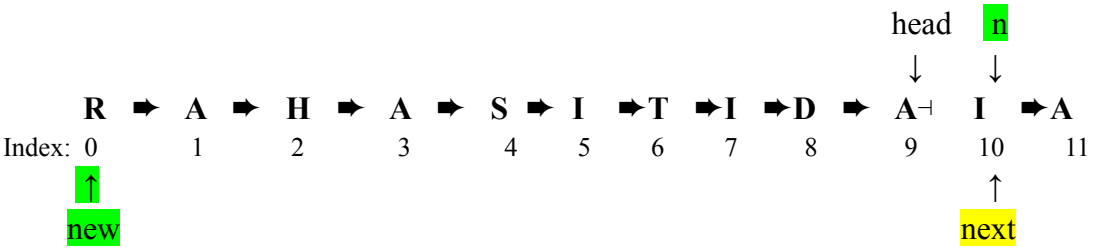
Again,



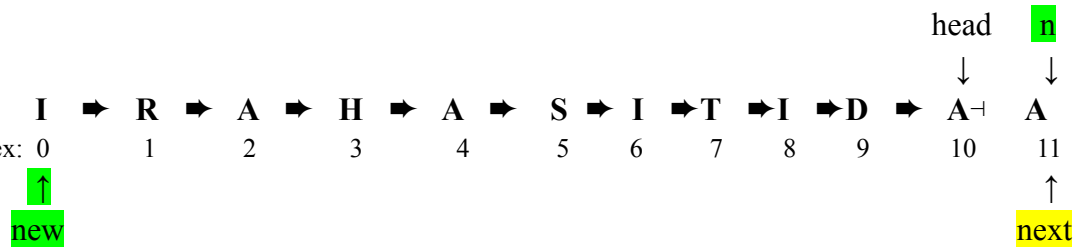
Again,



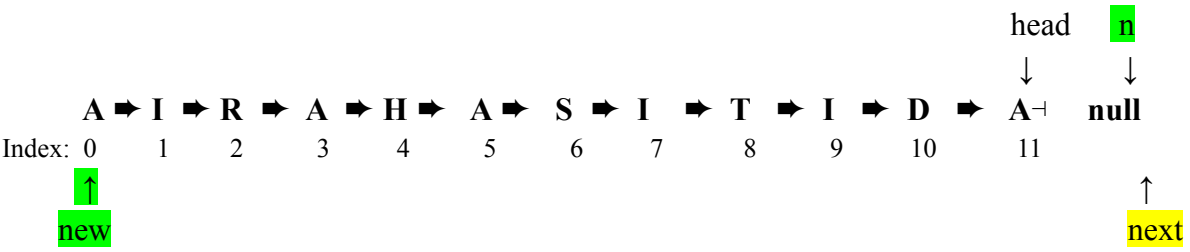
Again,



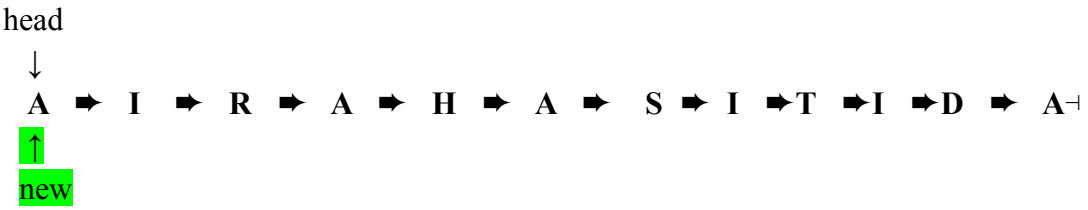
Again,

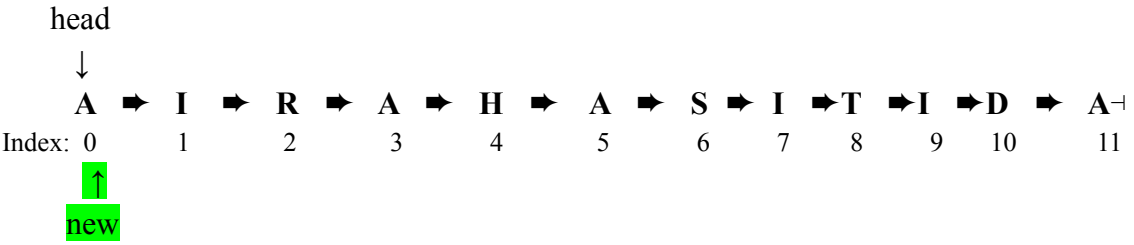


Again,

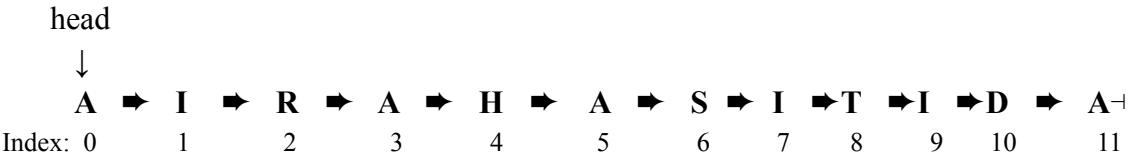


Last step:
head = new

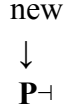


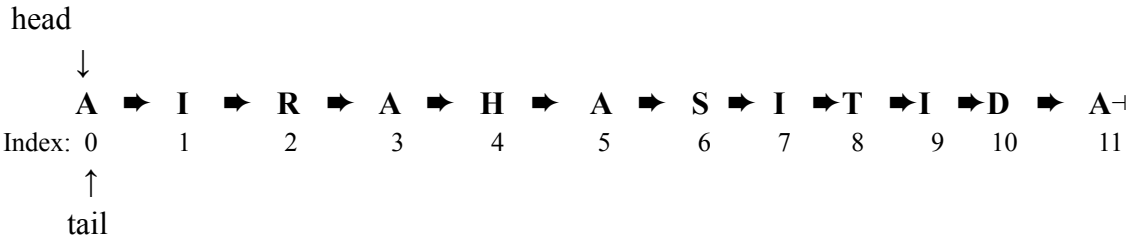


B.

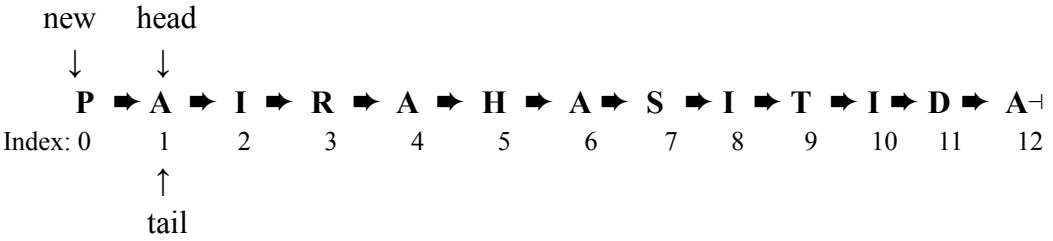


Creating new Node for P,
new = Node('P',None)
tail = head

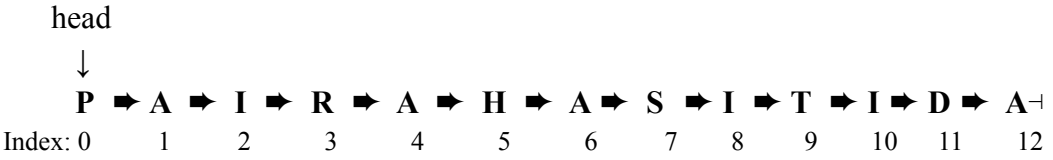




new.next = tail

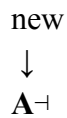


head = new

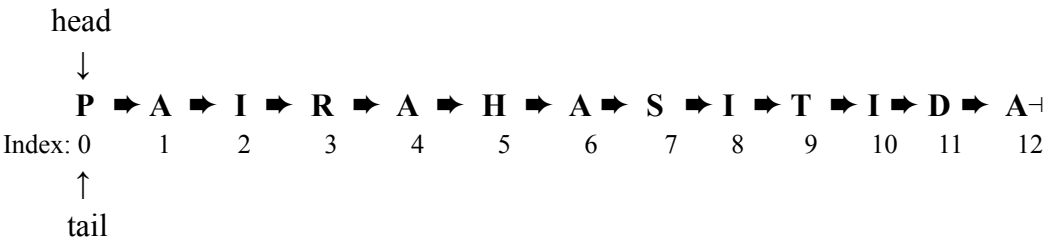


C.

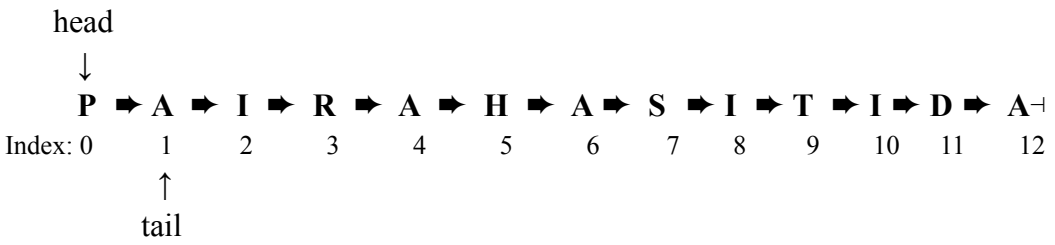
new = Node('A', None)



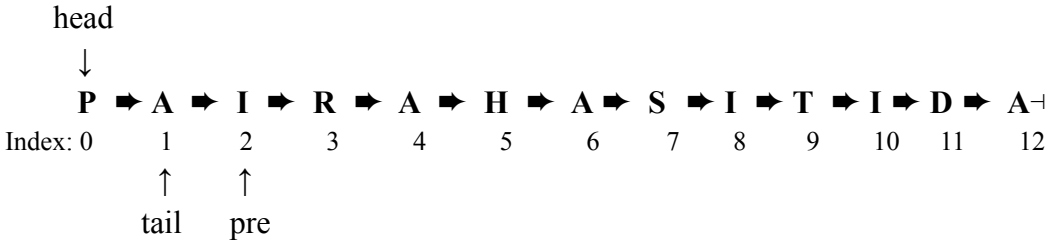
tail = head

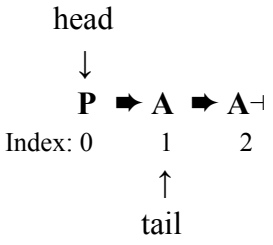


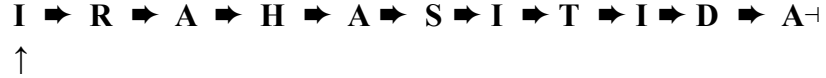
position = 2
for i in range(position-1):
    tail = tail.next
After the loop ends

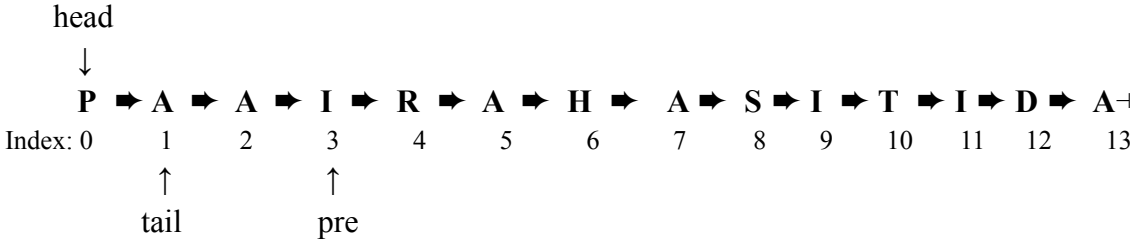


pre = tail.next
tail.next = new
new.next = pre







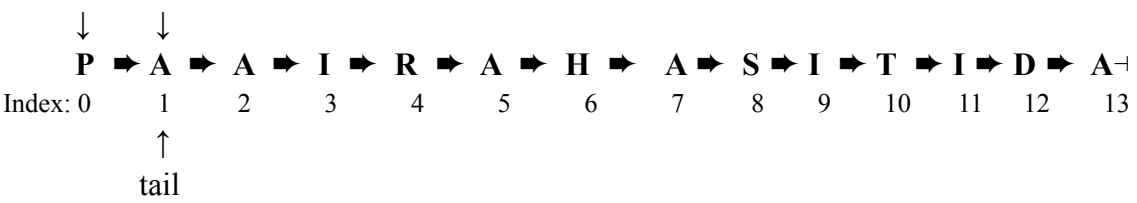


D.

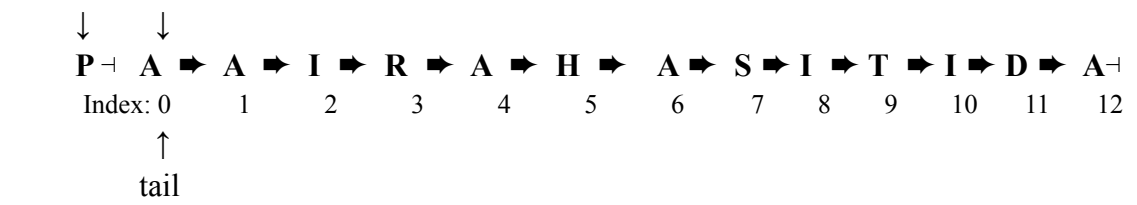
```
for i in range(4):
    #loop will run 4 times
```

Time 1:

```
    old_head = head
    head = head.next
    tail = head
```

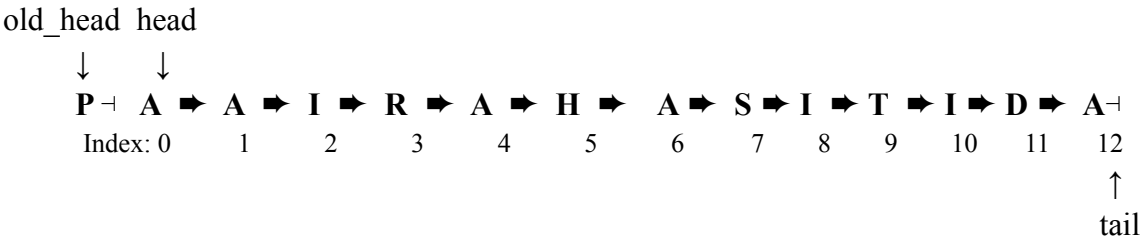



```
old_head.next = None
```

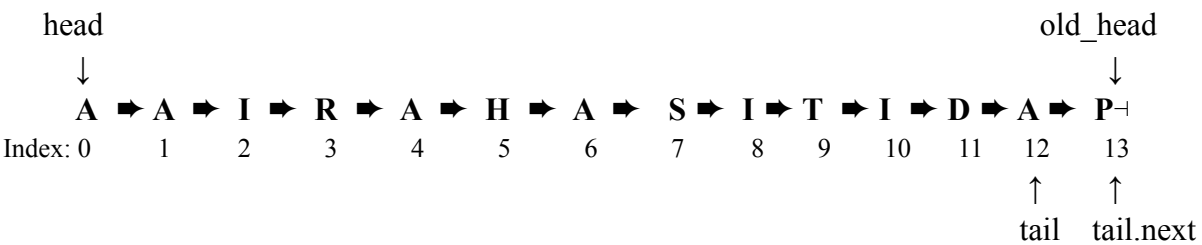



```
while tail.next is not None:
      tail = tail.next
```

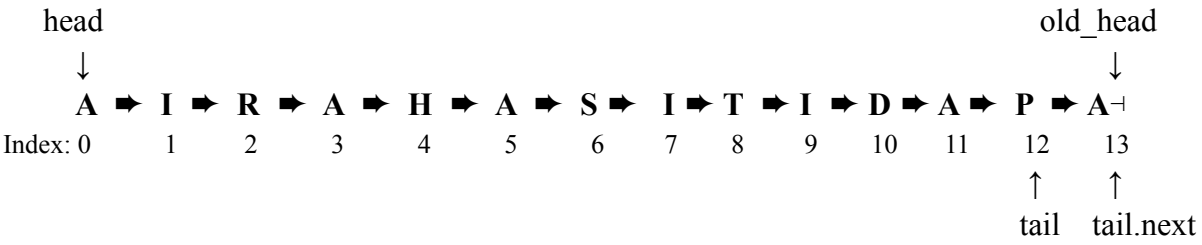
After the end of the loop tail will be index 12



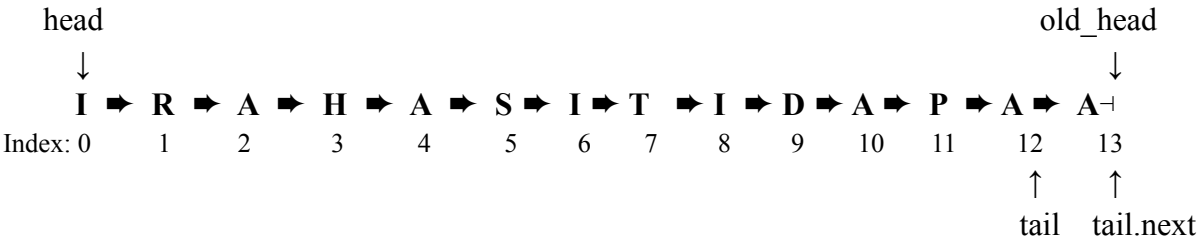

```
tail.next = old_head
```



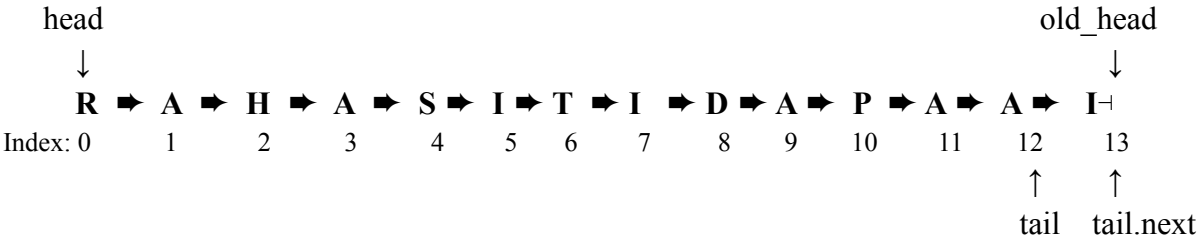
Time 2:



Time 3:

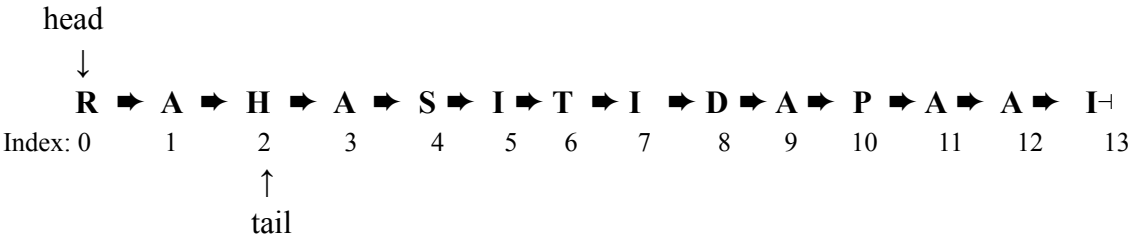


Time 4:

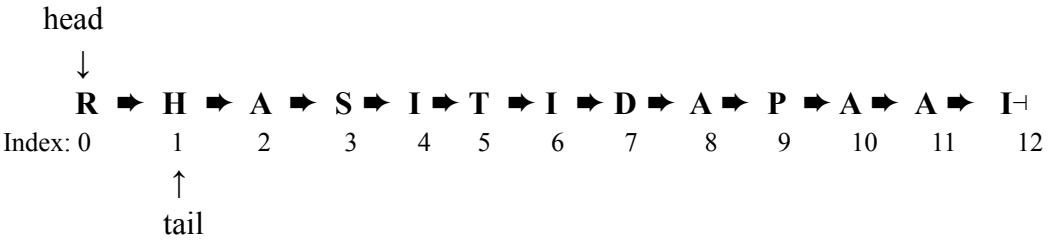


E.

Second elements index is 1
tail = head.next.next

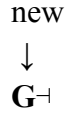


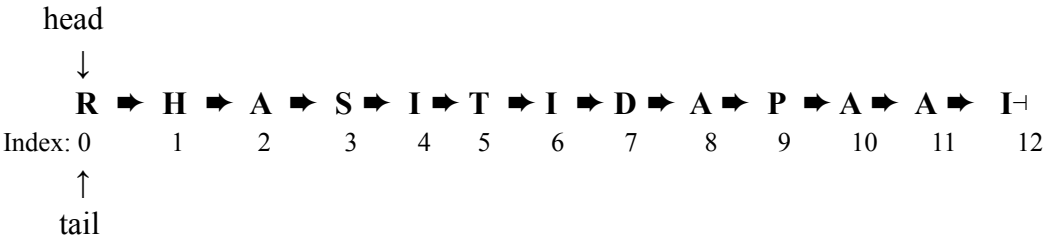
del = head.next
del.next = None
del.element = None
head.next = tail



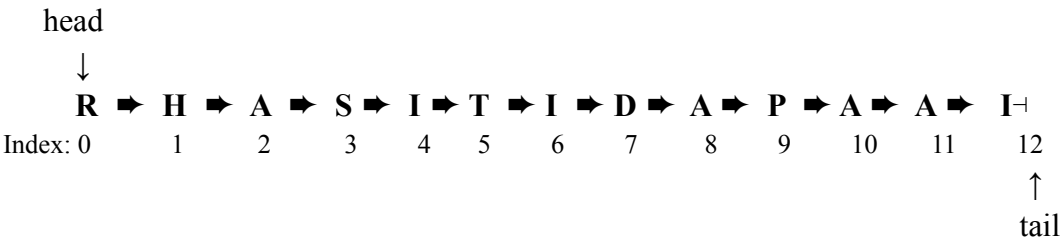
F

new = Node('G', None)
tail = head

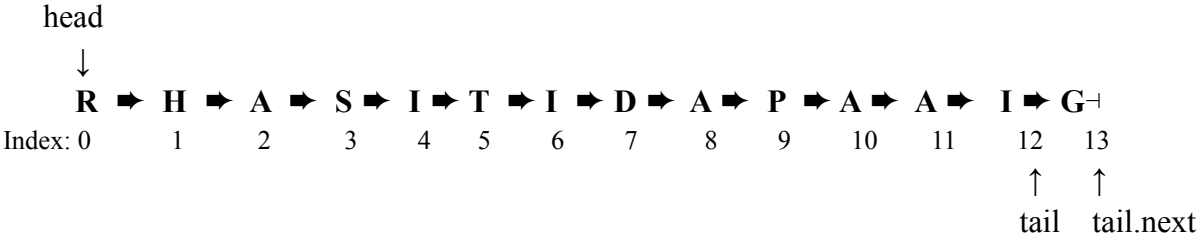





```
while tail.next is not None:
    tail = tail.next
```



tail.next = new



G.

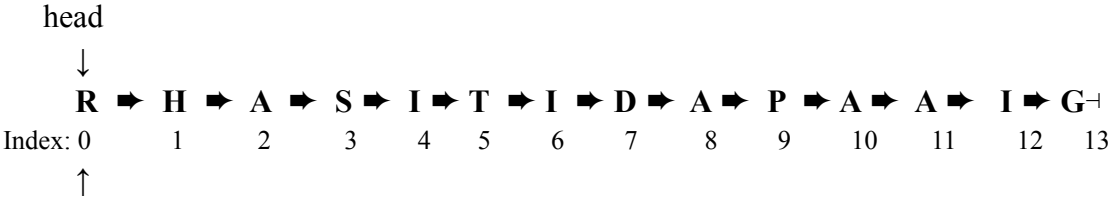
for i in range(3):

    #loop will run 3 times

Time 1:

tail = head

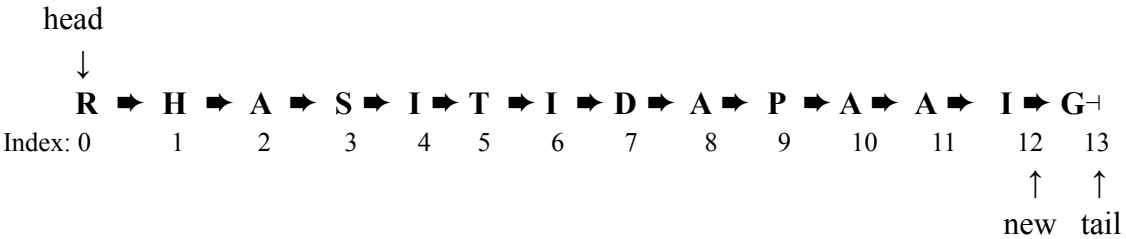
new = None



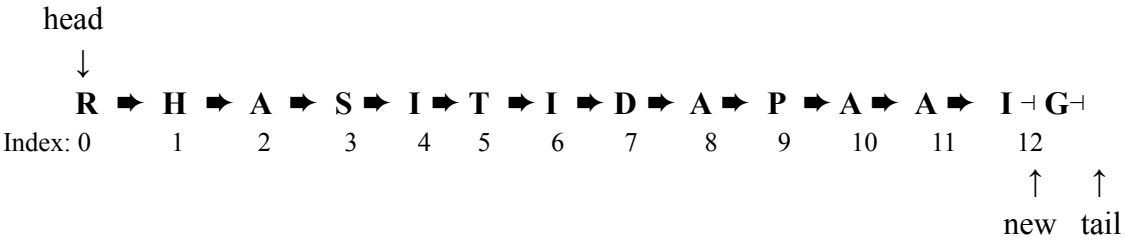
while tail.next is not None:

        new = tail

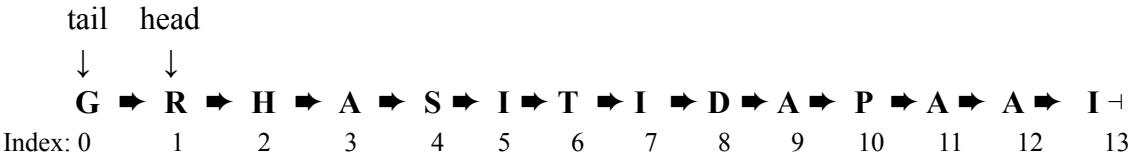
        tail = tail.next



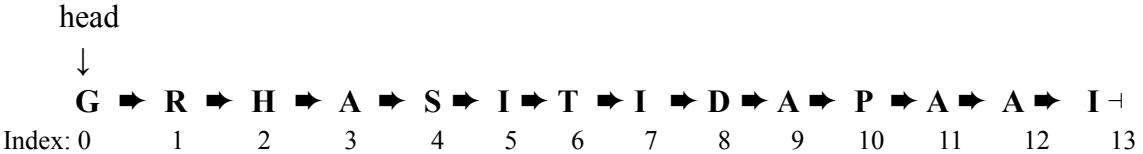
new.next = None



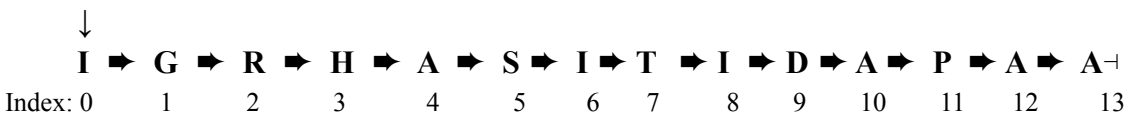
tail.next = head



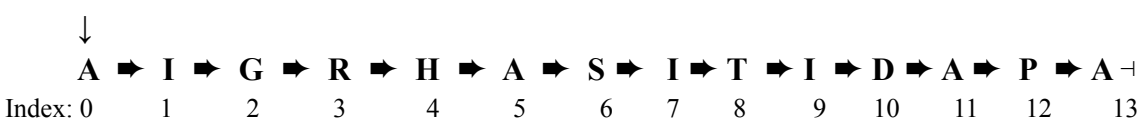
head = tail



Time 2:
Repeating the same steps



Time 3:
Repeating the same steps



H.

```
new = None
count = 0
tail = head
while tail is not None:
       count = count+1
       tail = tail.next

count = count+1
for i in range(count):
     #loop will run 13 times
inside the loop
```

Answer to Question No.4

```
def printDuplicate(head):
  tail = head
  while tail is not None:
     n = tail.next
     while n is not None:
        if n.element is tail.element:
           print(n.element)
           return
```

```
            else:
                n = n.next
        tail = tail.next
```

Answer to Question No.5

```
def remove_multiple_of_five(head):
    n = head
    while n is not None:
        tail = head
        new = None

        while tail is not None and tail.element%5!=0:
            new = tail
            tail = tail.next
        if new is None:
            head = tail.next
            n = head
        elif tail is None:
            new.next = None
        else:
            new.next = tail.next
        n = n.next
    if n is None and head.element%5==0:
        head = None

    return head
```

Answer to Question No.6

```
def sum(self):
    a = self.first_head.next
    b = self.scnd_head.next
    count1 = 0
    count2 = 0
    while a is not None:
        count1 += 1
        a = a.next
    a = self.first_head.next
    while b is not None:
```

```
        count2 += 1
        b = b.next
    b = self.scnd_head.next
    r1 = 0
    while a is not None:
        r1 = r1 + a.element*(10**(count1-1))
        count1 = count1-1
        a = a.next
    r2 = 0
    while b is not None:
        r2 = r2 + b.element*(10**(count2-1))
        count2 = count2-1
        b = b.next

    add = r1+r2
    print(add)
```

Answer to Question No.7

```
def insert(head, newElement, index=None):
    inserted_value = Node(newElement, None)
    copy_tail = head
    for i in range(index - 1):
        copy_tail = copy_tail.next
    pre = copy_tail
    inserted_value.next = pre.next
    pre.next = inserted_value

    return head
```

Answer to Question No.8

```
def insert(head, elem, newElement):
    box = Node(newElement, None, None)
    tail = head.next
    new = None
    while tail.val is not elem and tail is not None:
        new = tail
        tail = tail.next
    pre = new
    box.next = pre.next
    pre.next.prev = box
    box.prev = pre
    pre.next = box

    return head
```