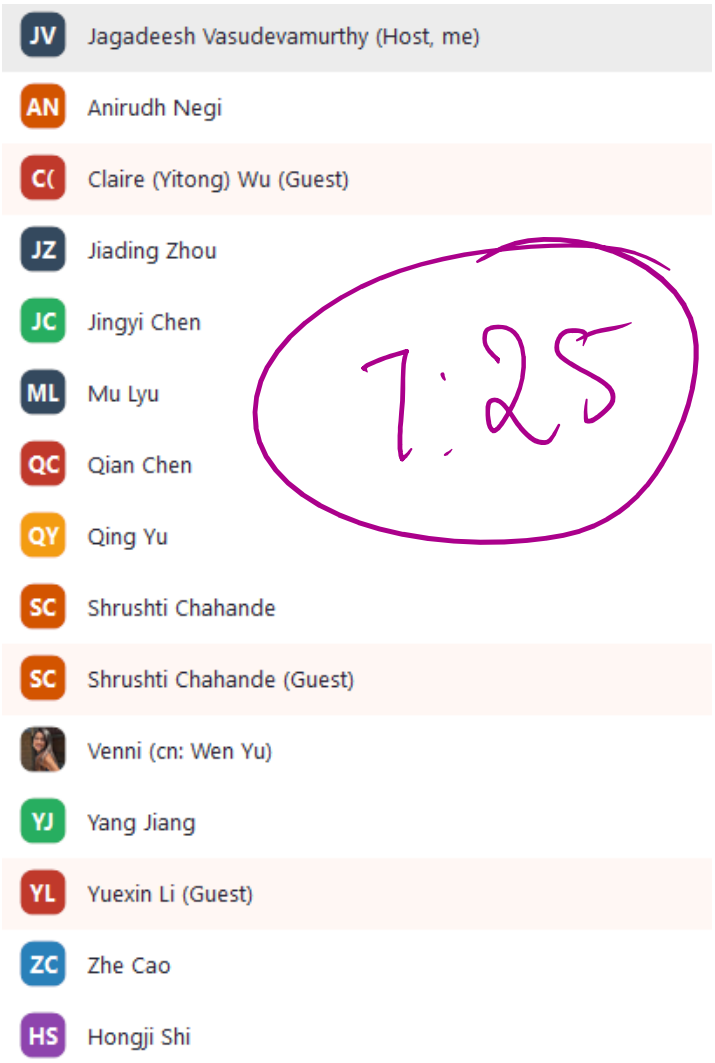
DAA PY 5

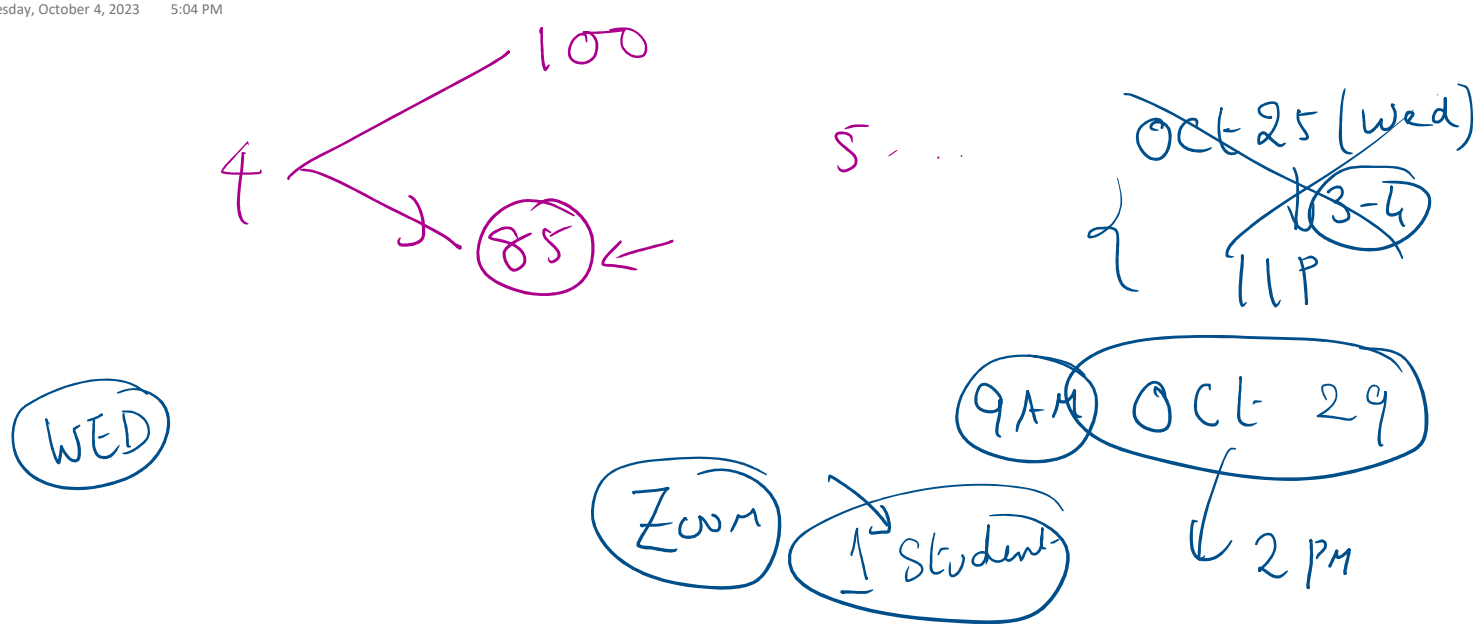
Wednesday, October 4, 2023 4:43 PM

5 PM



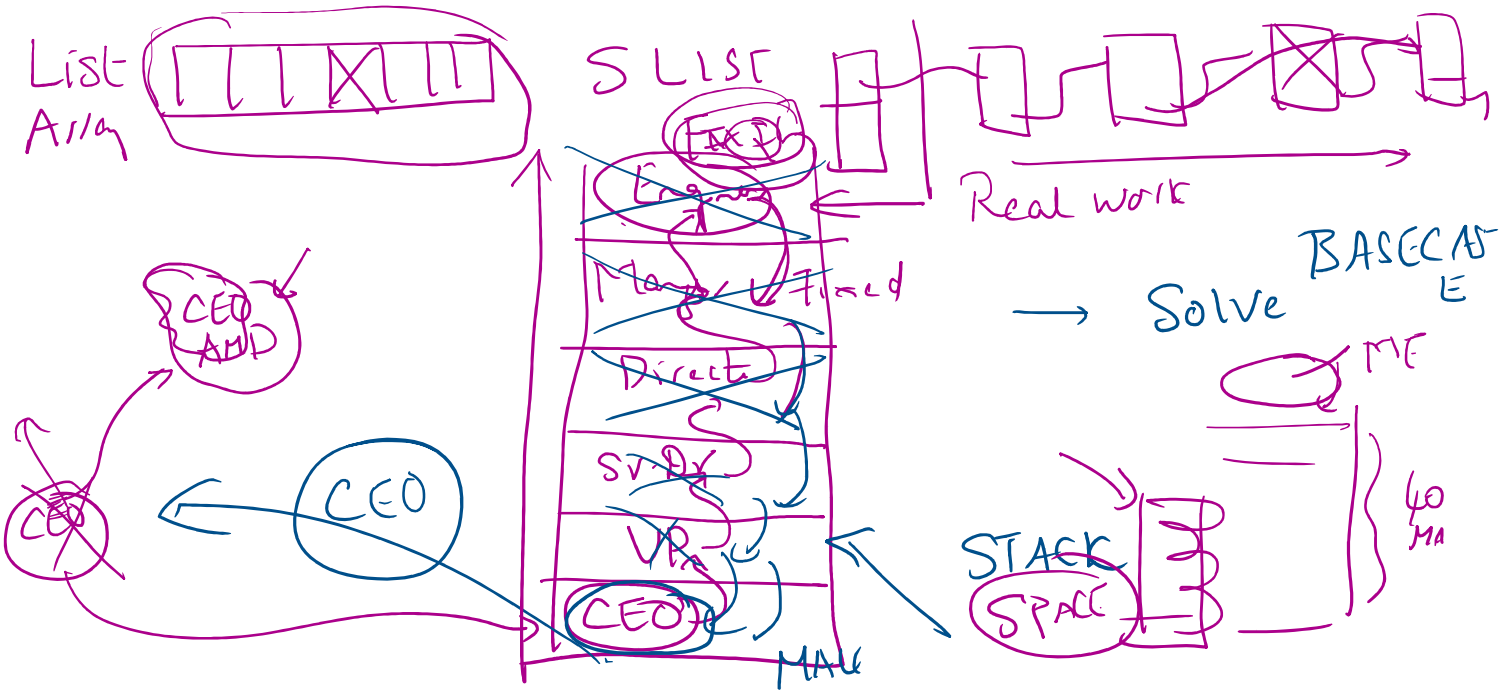
1

Wednesday, October 4, 2023 5:04 PM

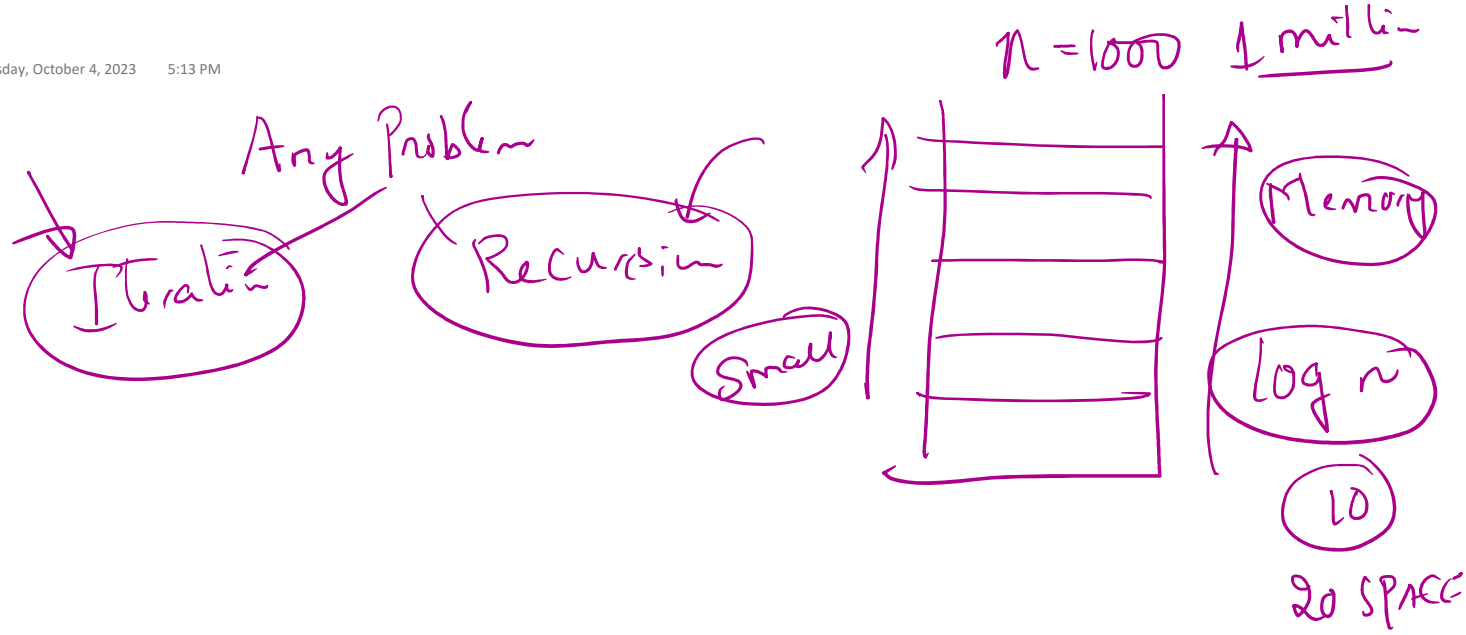


Wednesday, October 4, 2023 5:05 PM

Recursion

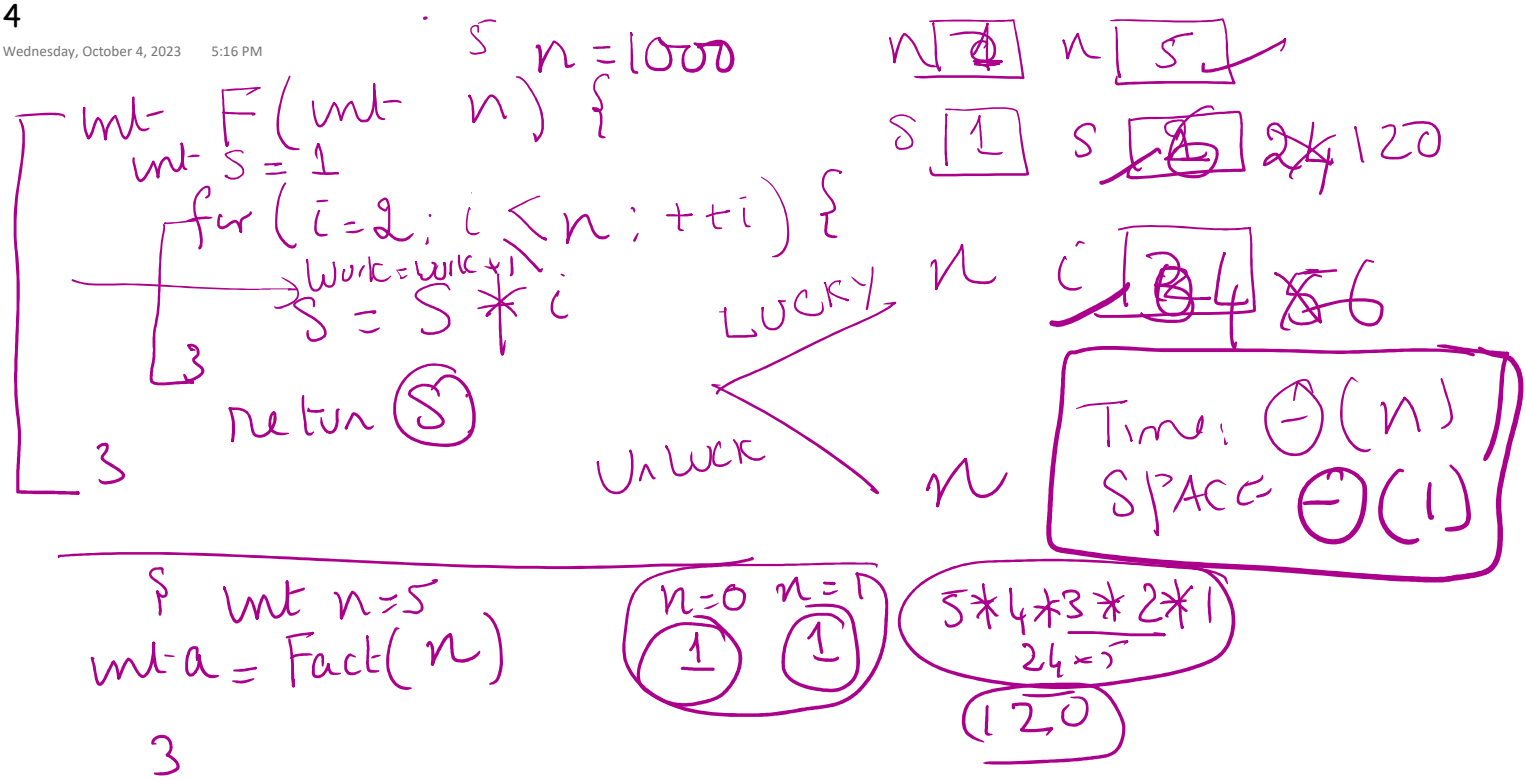


Wednesday, October 4, 2023 5:13 PM



4

Wednesday, October 4, 2023 5:16 PM



Wednesday, October 4, 2023 5:21 PM

```
int F(int n)
{
  if(n<=2)
    return 1;
  return(n* F(n-1));
}
```

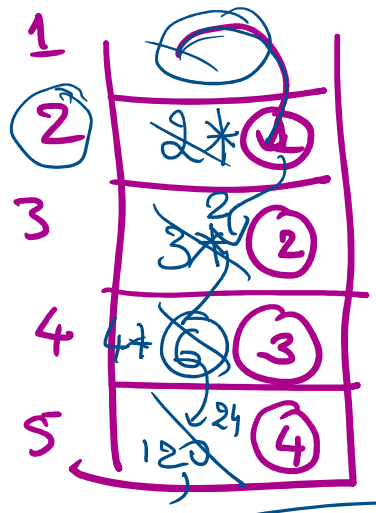
1! = 1 = 1

2! = 2 * 1 = 2

3! = 3 * 2 = 6

4! = 4 * 6 = 24

5! = 5 * 24 = 120



MULTIPLICATION

```
{
  int n=5;
  int a=F(n)
}
```

a = 120

Wednesday, October 4, 2023 5:26 PM

n = 1000

n = 5

```
int F(int n) {
    if(n <= 2)
        return 1
    return (n * F(n-1));
}
```

Time Complexity

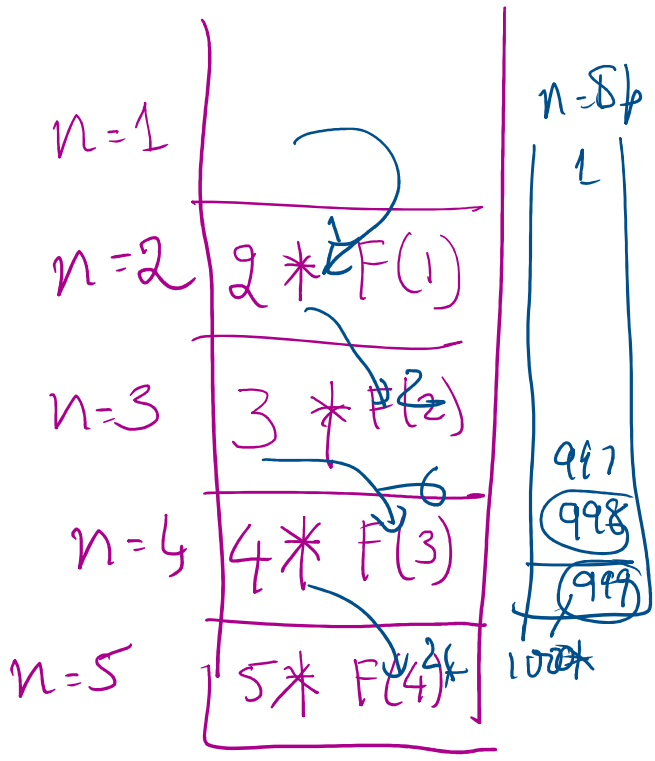
3

{ int n
int a = F(n)
return a

0	1	5
1	1	120

120

3



7

Wednesday, October 4, 2023 5:30 PM

$T(n) = T(n-1) + C$

$T(1) = 1$

$T(0) = 0$

Time $= Cn$

$= \Theta(n)$

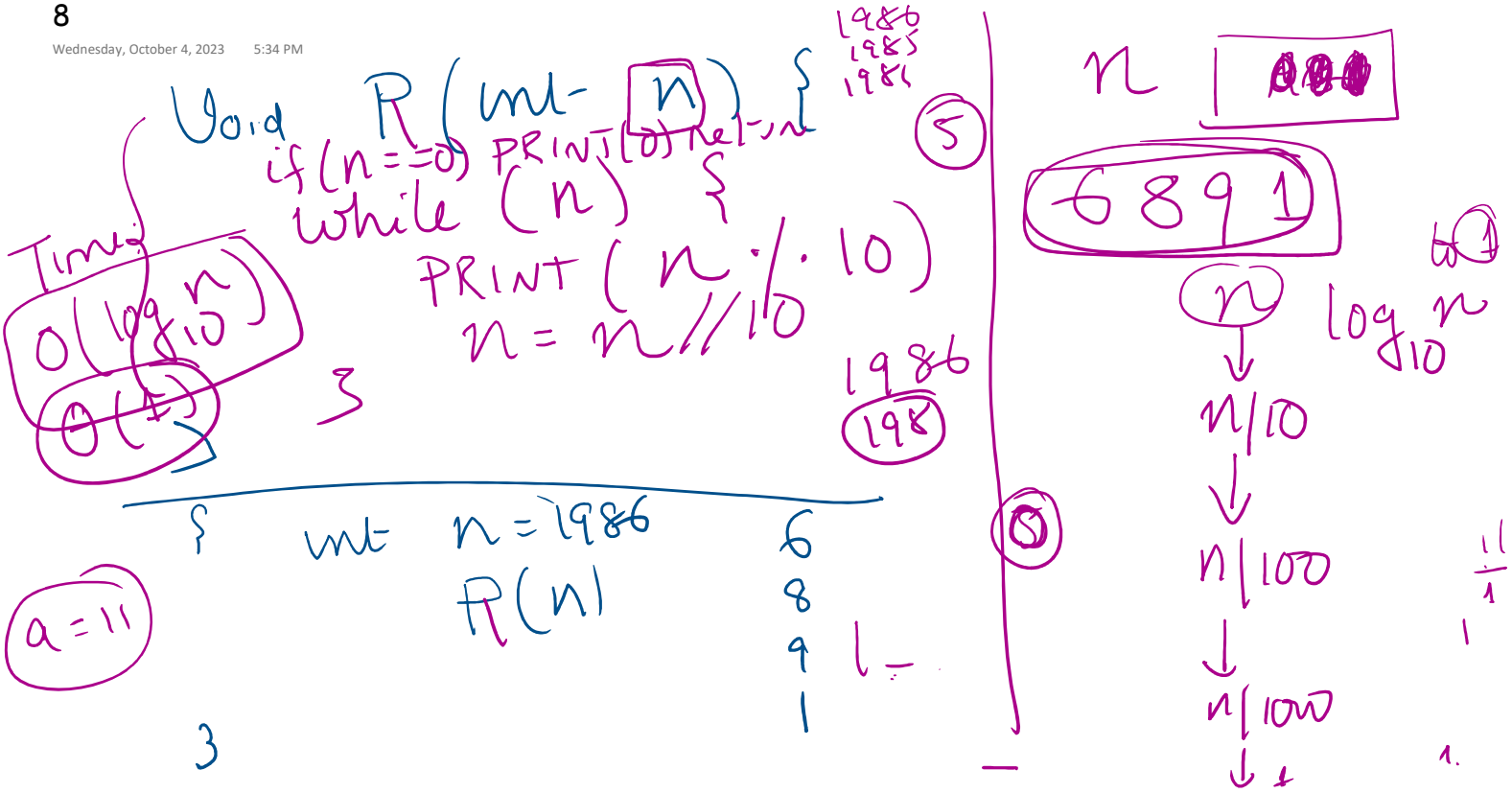
SPACE $= \Theta(n)$

~~n~~	C	T
~~n-1~~	C	|
↓		n
~~n-2~~	C	|
↓		|
n-3		|
↓		|
n=1	C	

5 * 4

8

Wednesday, October 4, 2023 5:34 PM



9

Wednesday, October 4, 2023 5:50 PM

```
void R(int n) {
    if (n < 10)
        PRINT(n)
    else
        R(n//10)
        PRINT(n%10)
}

{
    int n
    R(n)
}
```

1986 → 198 → 19 → 1

n = 1	R(1)
n = 19	P(19 % 10)
n = 198	R(19)
	P(198 % 10)
n = 1986	R(198)
	PRINT(1986 % 10)

1 9 8 6

0 1

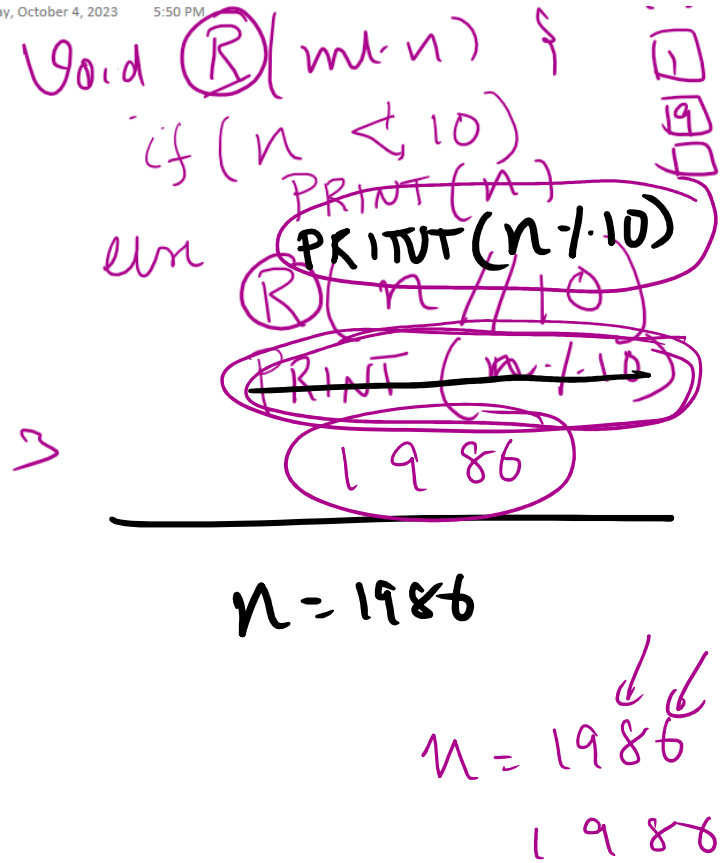
0 1

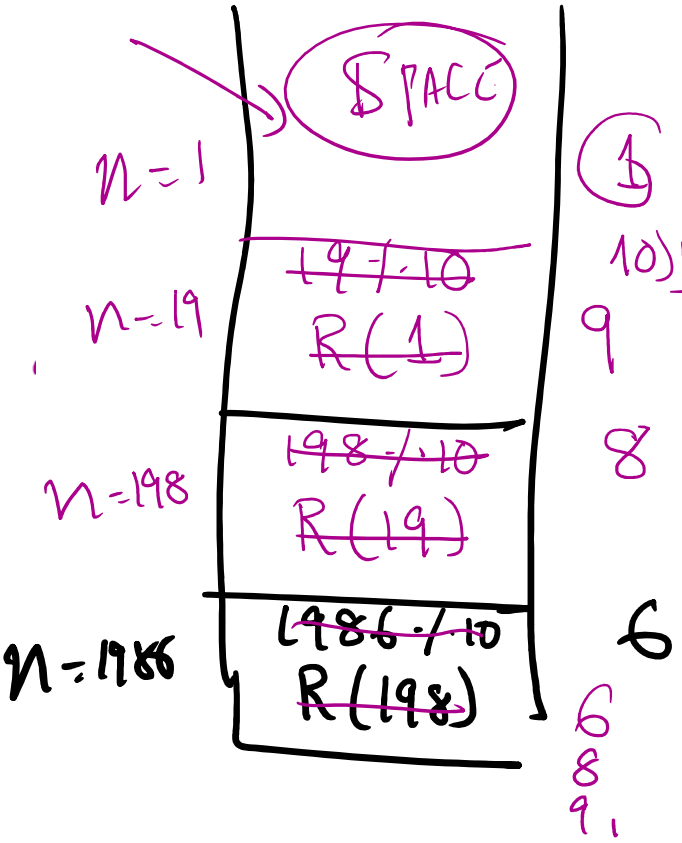
3

10

Wednesday, October 4, 2023 5:58 PM

iesday, October 4, 2023 5:50 PM

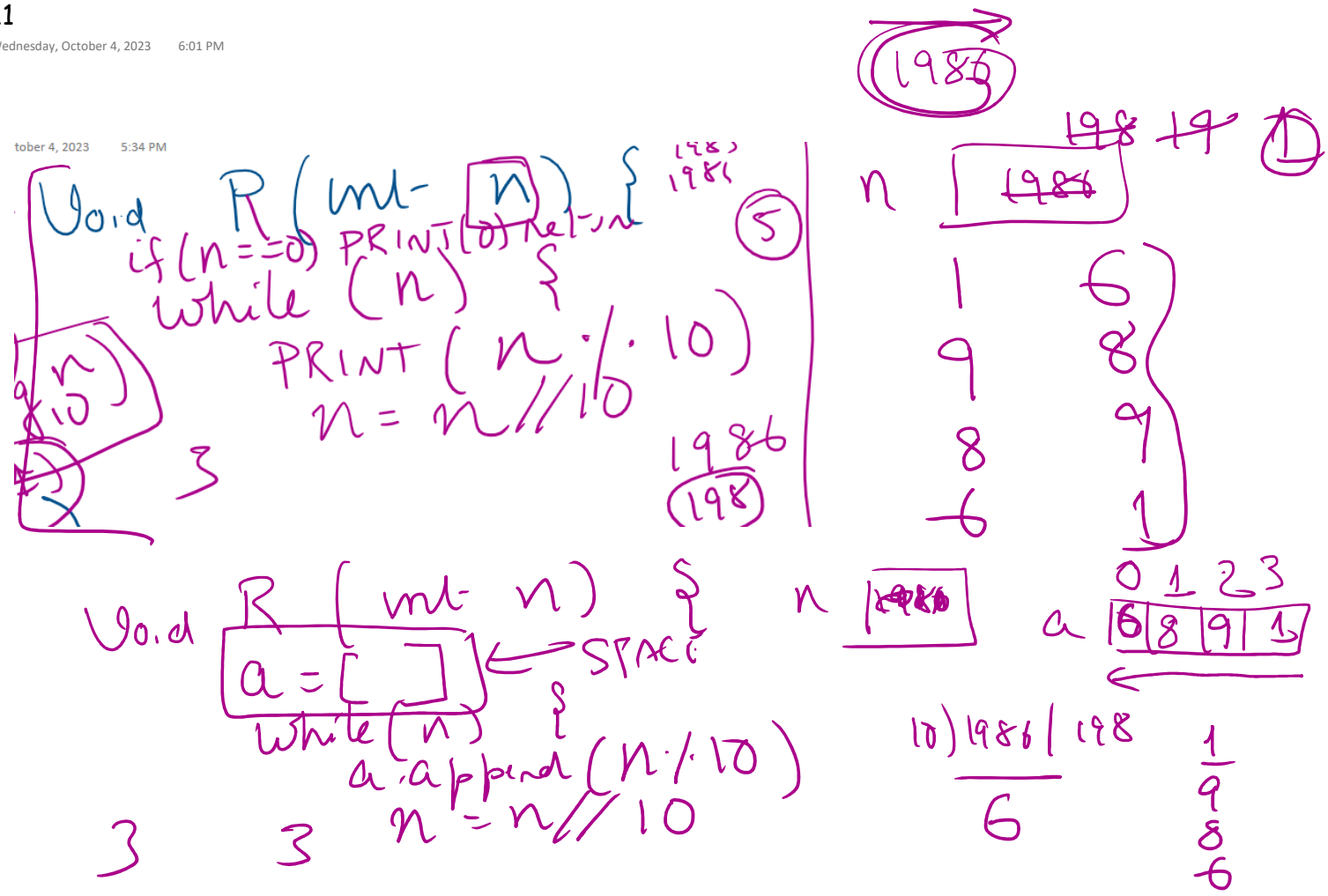




n = 1986

1986

Wednesday, October 4, 2023 6:01 PM



12

Wednesday, October 4, 2023 6:06 PM

$T(n) = T(n/10) + c$ — $\log_{10} n$

$T(0..9) = 1$

$T(n) : T(n+1)$

$\log_{10} n$

SPACE

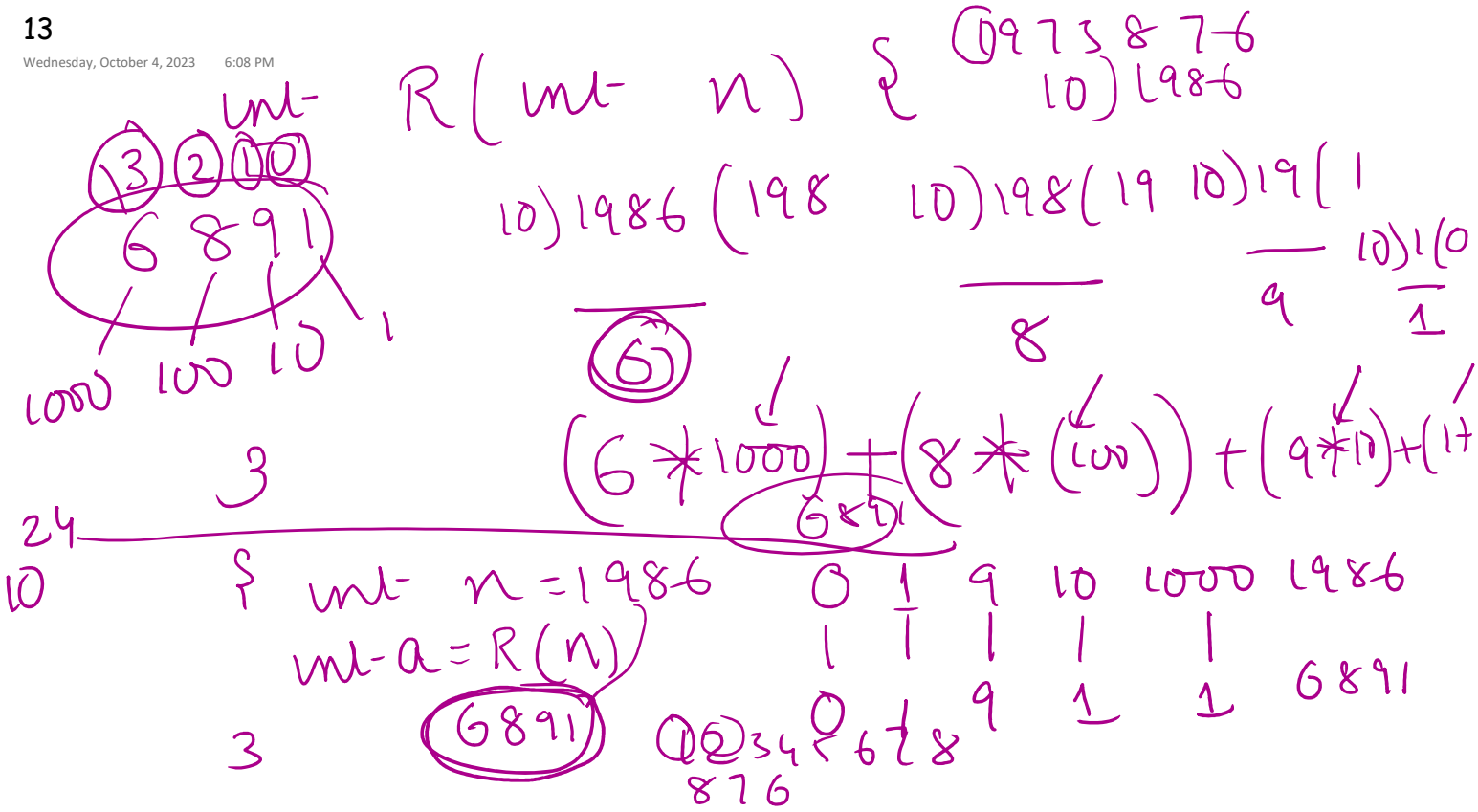
$\log n$

SPACE OF The stack

$\log n$ — 20, 30

13

Wednesday, October 4, 2023 6:08 PM



Wednesday, October 4, 2023 6:13 PM

int- R(int- n)

10)1911
9

10)198(19)
8

10)1986
6

3

10)1(0
1

{ int- n = 1000 1986
a = R(n)
3 (1) 6891

a) 0 { 1 2 3
1986
6 8 9 1

S = 0

S = S * 10 + a[i]

S 6891

Wednesday, October 4, 2023 6:16 PM

```
int R(int n){
    int s=0
    while(n!=0){
        s=s*10+(n%10)
        n=n//10
    }
    return s;
}
```

19867910

Time: log n

SPACE: 2

n 10000

s 0 1

10)1000(100

0

{ n=1986

int a=R(n)

3

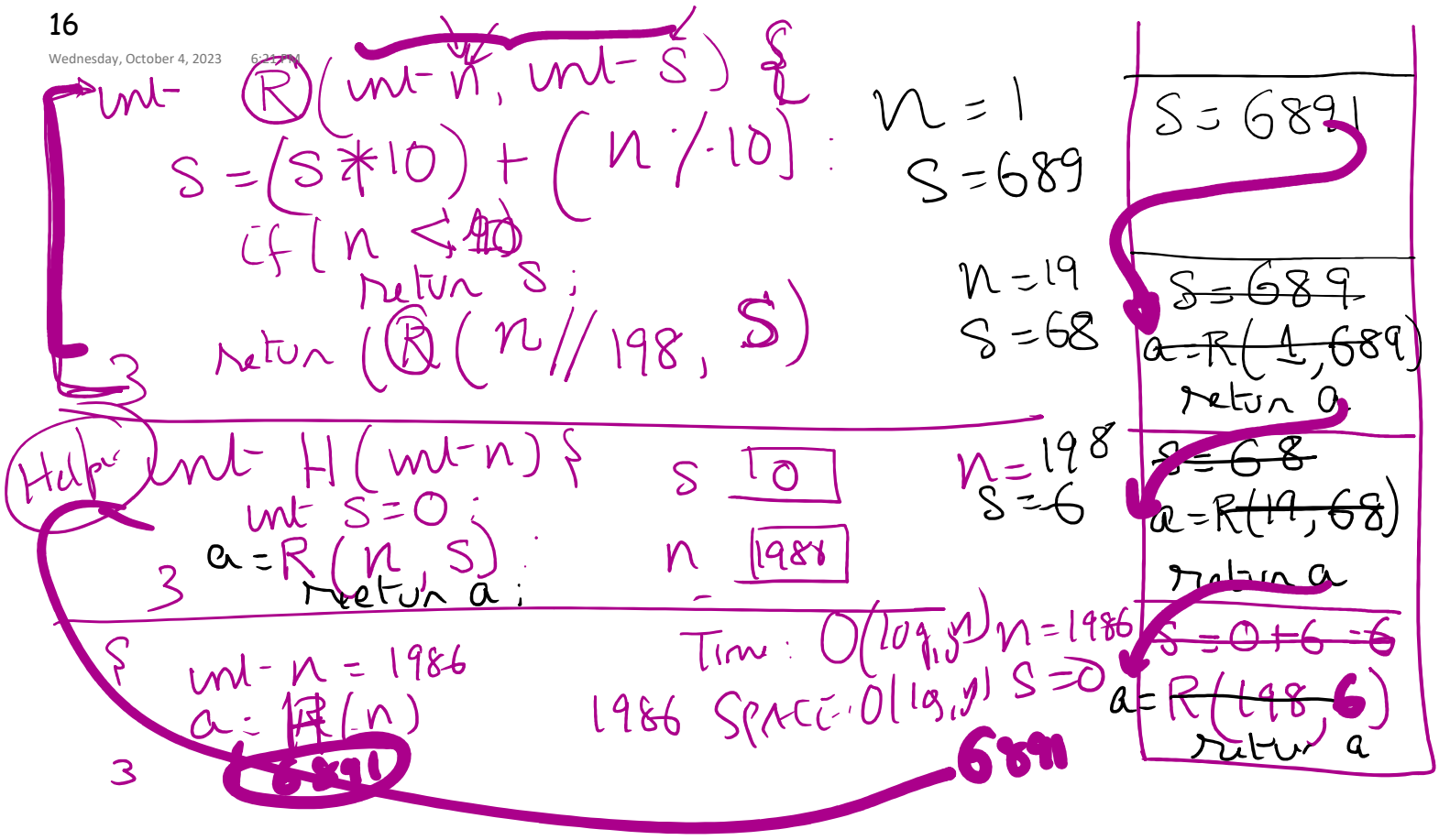
n=0 n=1

a=0

n=1986 n=1000

6891 1

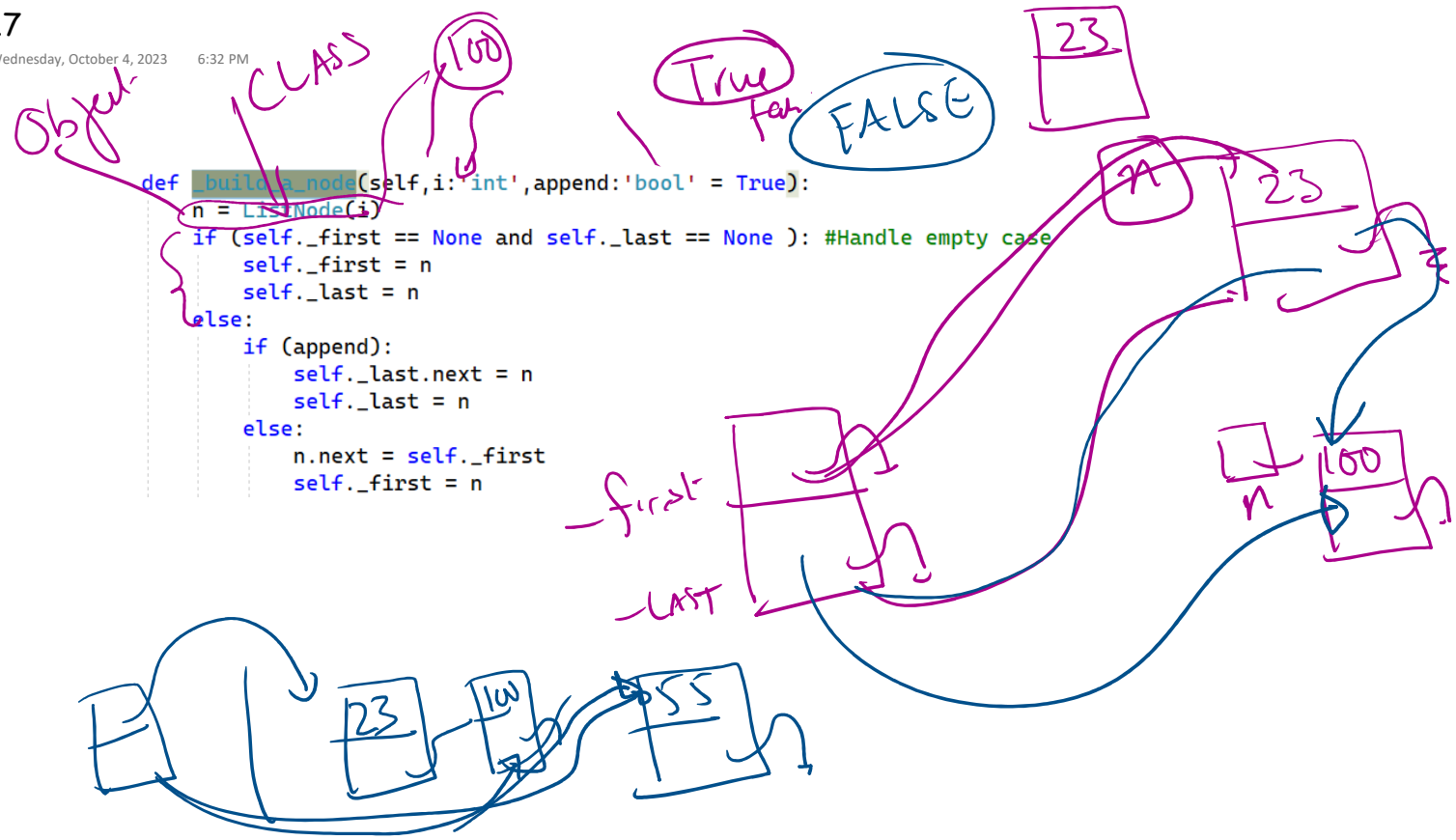
Wednesday, October 4, 2023 6:21 PM



17

Wednesday, October 4, 2023 6:32 PM

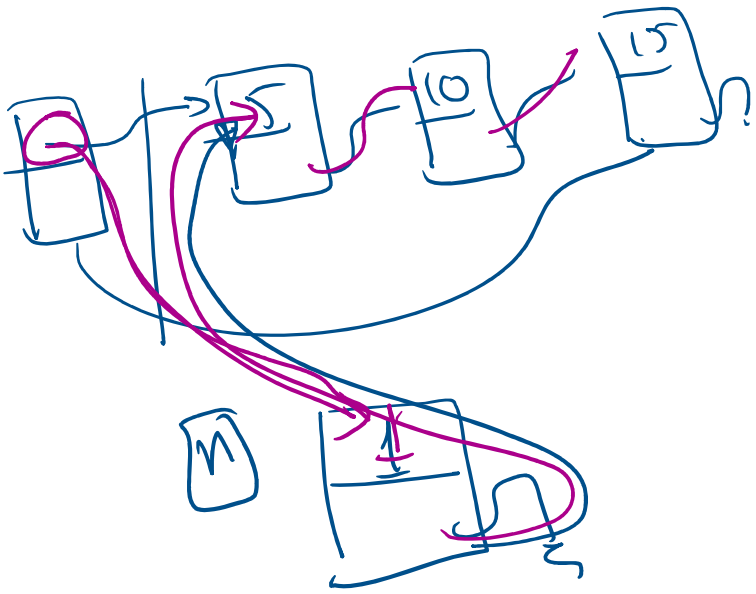
```
def _build_a_node(self,i:'int',append:'bool' = True):
    n = ListNode(i)
    if (self._first == None and self._last == None ): #Handle empty case
        self._first = n
        self._last = n
    else:
        if (append):
            self._last.next = n
            self._last = n
        else:
            n.next = self._first
            self._first = n
```



18

Wednesday, October 4, 2023 7:02 PM

```python
def _build_a_node(self,i:'int',append:'bool' = True):
    n = ListNode(i)
    if (self._first == None and self._last == None ): #Handle empty case
        self._first = n
        self._last = n
    else:
        if (append):
            self._last.next = n
            self._last = n
        else:
            n.next = self._first
            self._first = n
```



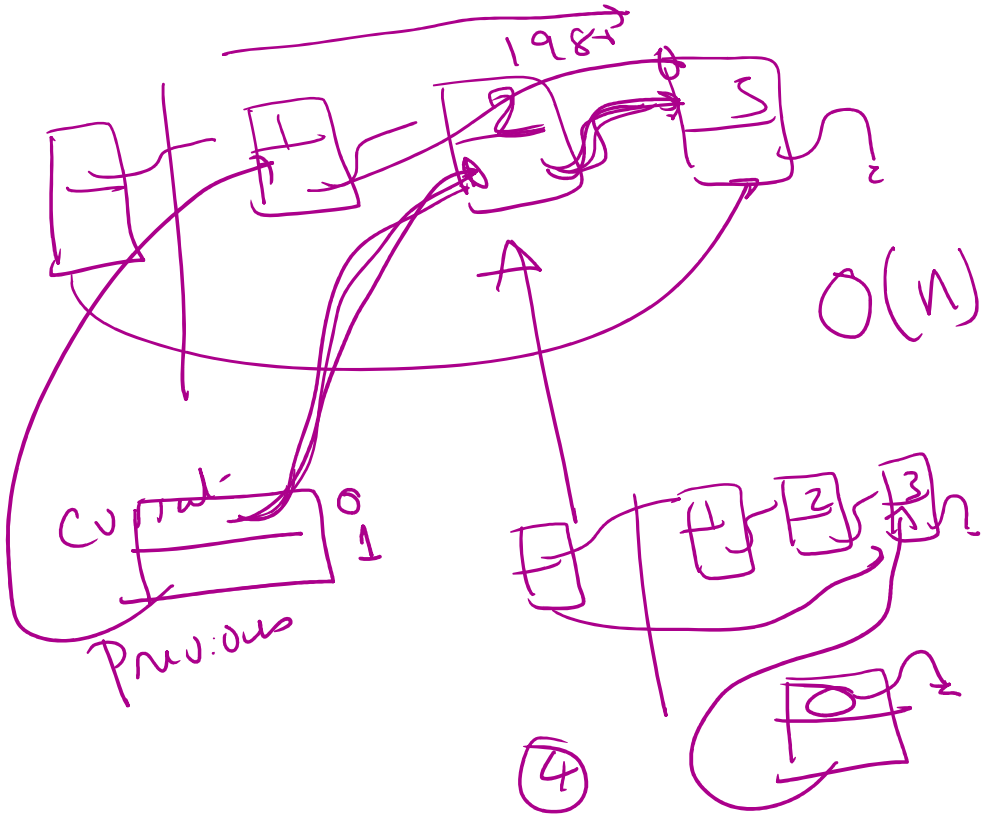
19

Wednesday, October 4, 2023 7:04 PM

```
def find(self,x:'int'):
    nodes = self._find(x)
    if (nodes[0]):
        return True
    else:
        return False
```

O(n)

Delete

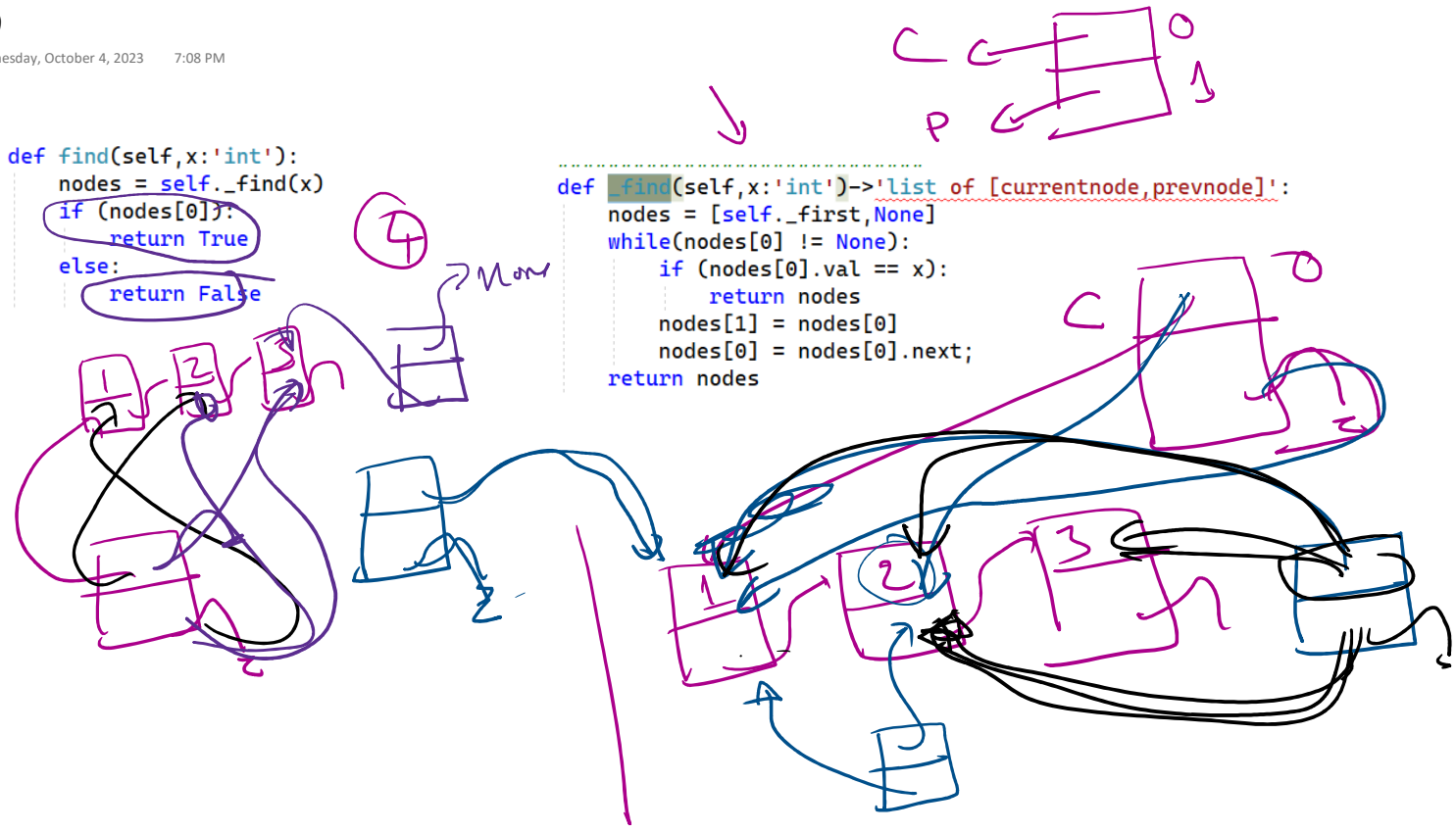


20

Wednesday, October 4, 2023 7:08 PM

```
def find(self,x:'int'):
    nodes = self._find(x)
    if (nodes[0]):
        return True
    else:
        return False
```

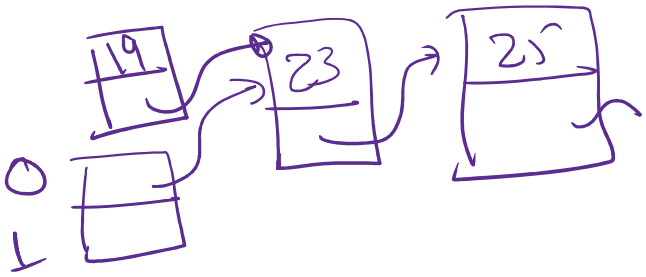
```
def _find(self,x:'int')->'list of [currentnode,prevnode]':
    nodes = [self._first,None]
    while(nodes[0] != None):
        if (nodes[0].val == x):
            return nodes
        nodes[1] = nodes[0]
        nodes[0] = nodes[0].next;
    return nodes
```



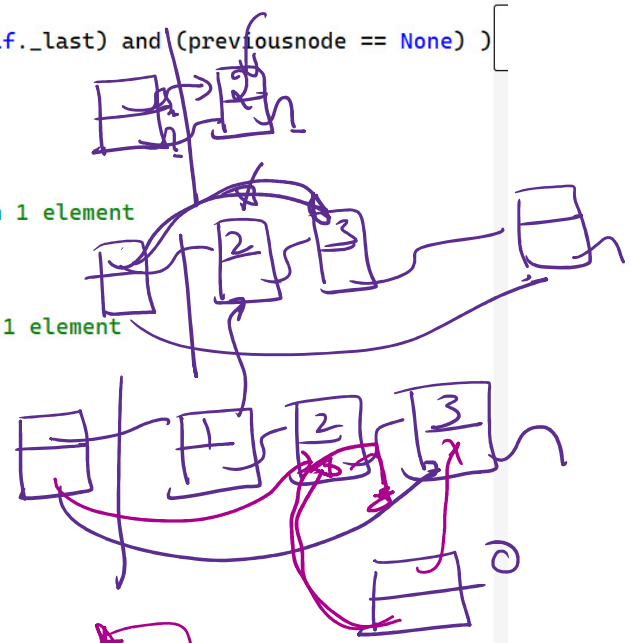
21

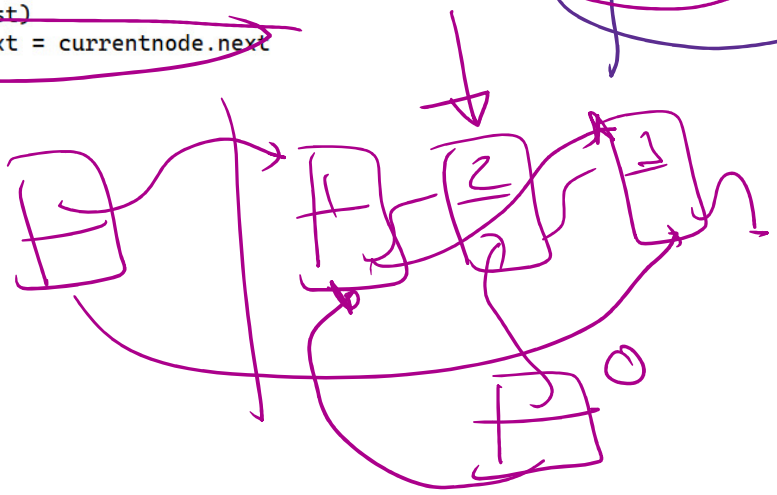
Wednesday, October 4, 2023 7:13 PM

```
def delete(self,x:'int')->'bool':
    nodes = self._find(x)
    a =  self._unhook(nodes)
    return a ;
```




```
##################################
def _unhook(self,nodes:'list of size 2')->'bool':
    if (nodes[0]):
        currentnode = nodes[0]
        previousnode = nodes[1]
        if ( (currentnode == self._first) and (currentnode == self._last) and (previousnode == None) ):
            ## list has only one element
            assert(self._first == self._last)
            self._first = None
            self._last  = None
        elif (currentnode == self._first):
            ##first element being removed and list has more than 1 element
            assert(self._first.next != None)
            self._first = currentnode.next
        elif (currentnode == self._last):
            ## last element being removed and list has more than 1 element
            assert(self._first)
            previousnode.next = None
            self._last = previousnode
        else:
            ## You are removing middle element
            assert(self._first)
            assert(self._last)
            previousnode.next = currentnode.next
        return True
    else:
        return False
```

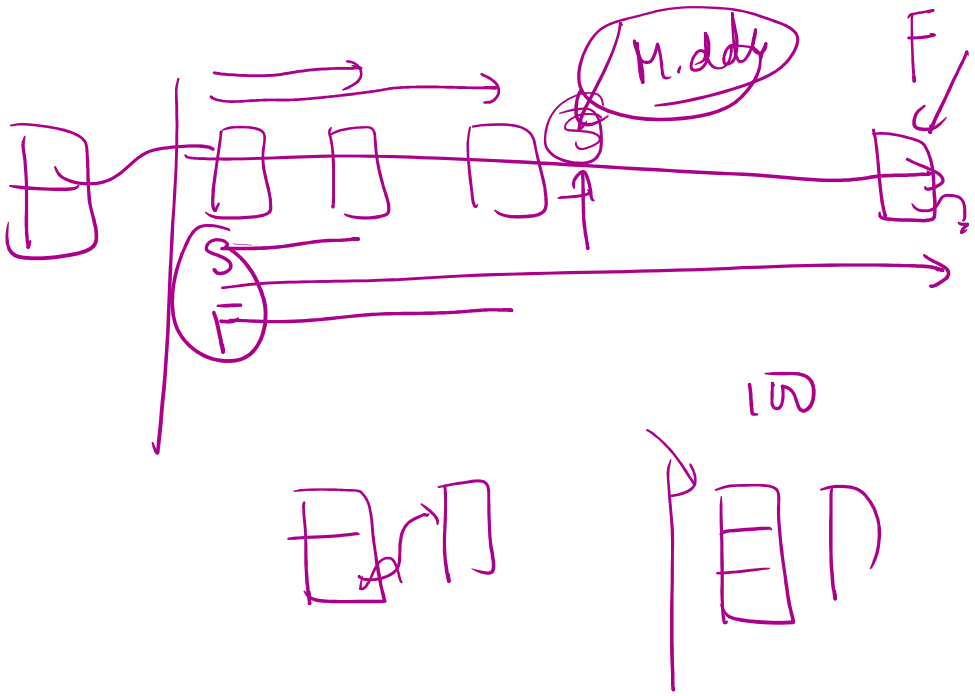




22

Wednesday, October 4, 2023 7:21 PM

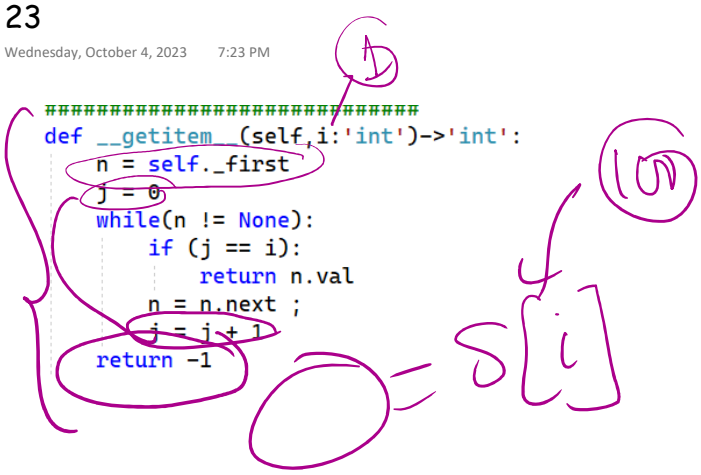
```
def find_mid_point(self)->'int':
    if (self._first == None):
        return 0
    s = self._first
    f = s.next
    while(f != None):
        f = f.next
        if (f != None):
            f = f.next
            s = s.next
    return s.val
```

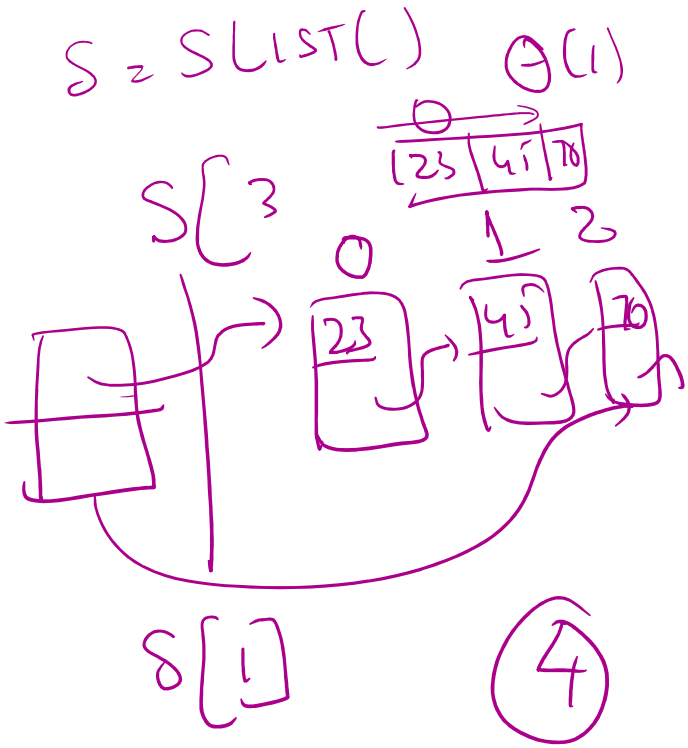


23

Wednesday, October 4, 2023 7:23 PM

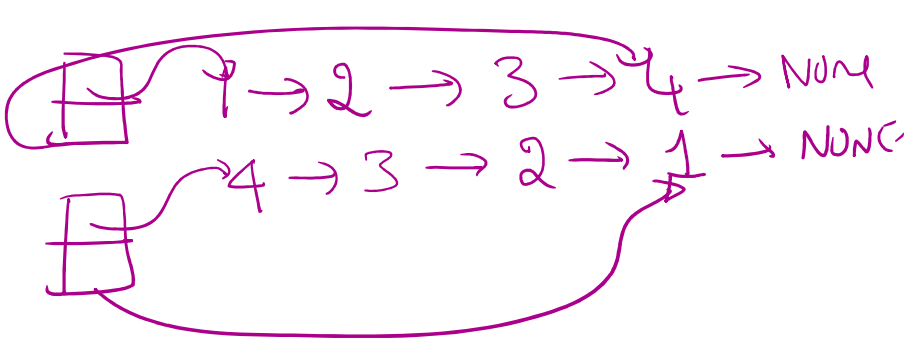
```
##############################
def __getitem__(self,i:'int')->'int':
    n = self._first
    j = 0
    while(n != None):
        if (j == i):
            return n.val
        n = n.next ;
        j = j + 1
    return -1
```

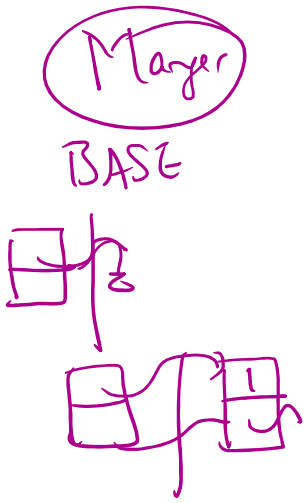




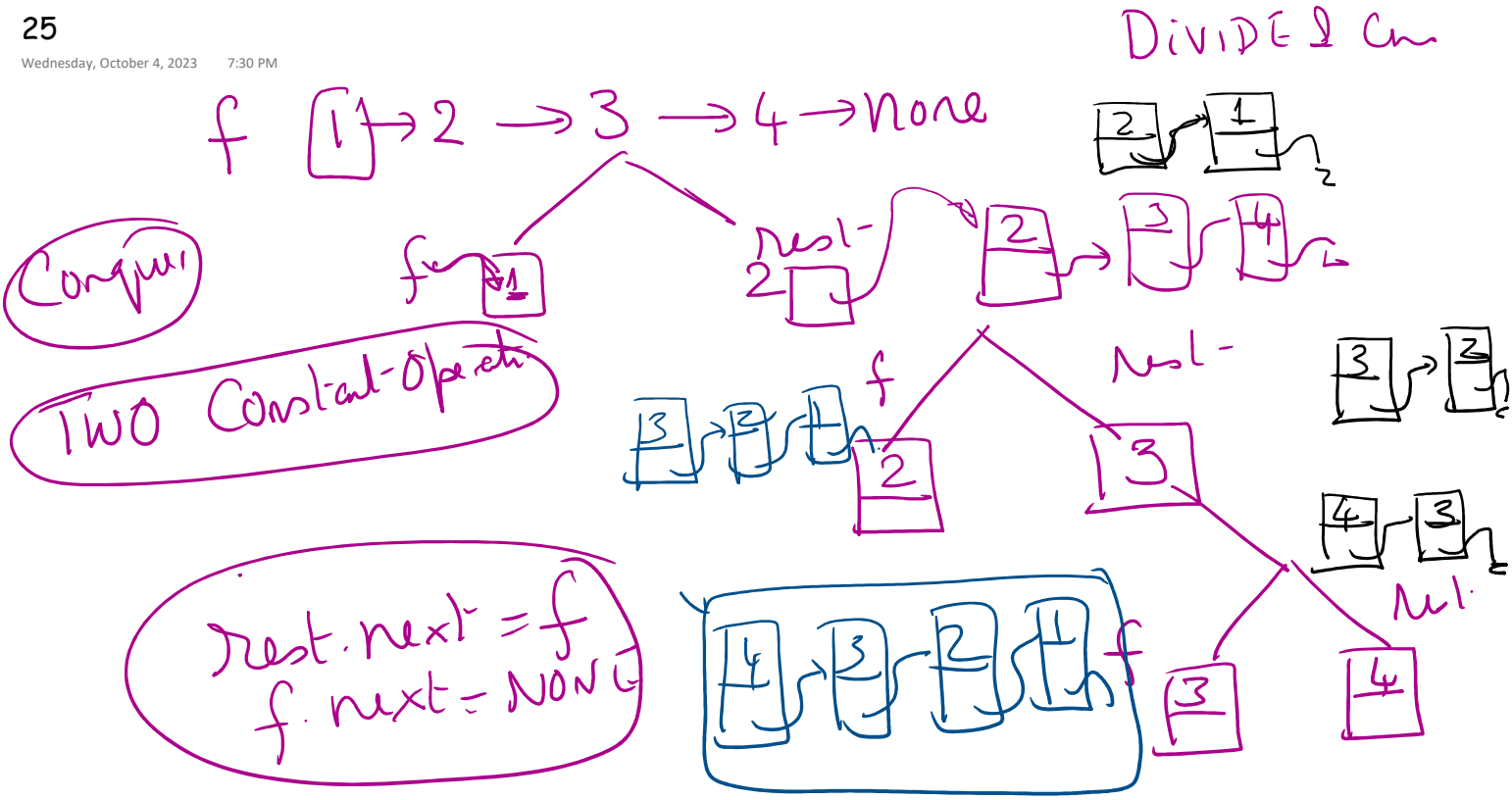
24

Wednesday, October 4, 2023 7:28 PM

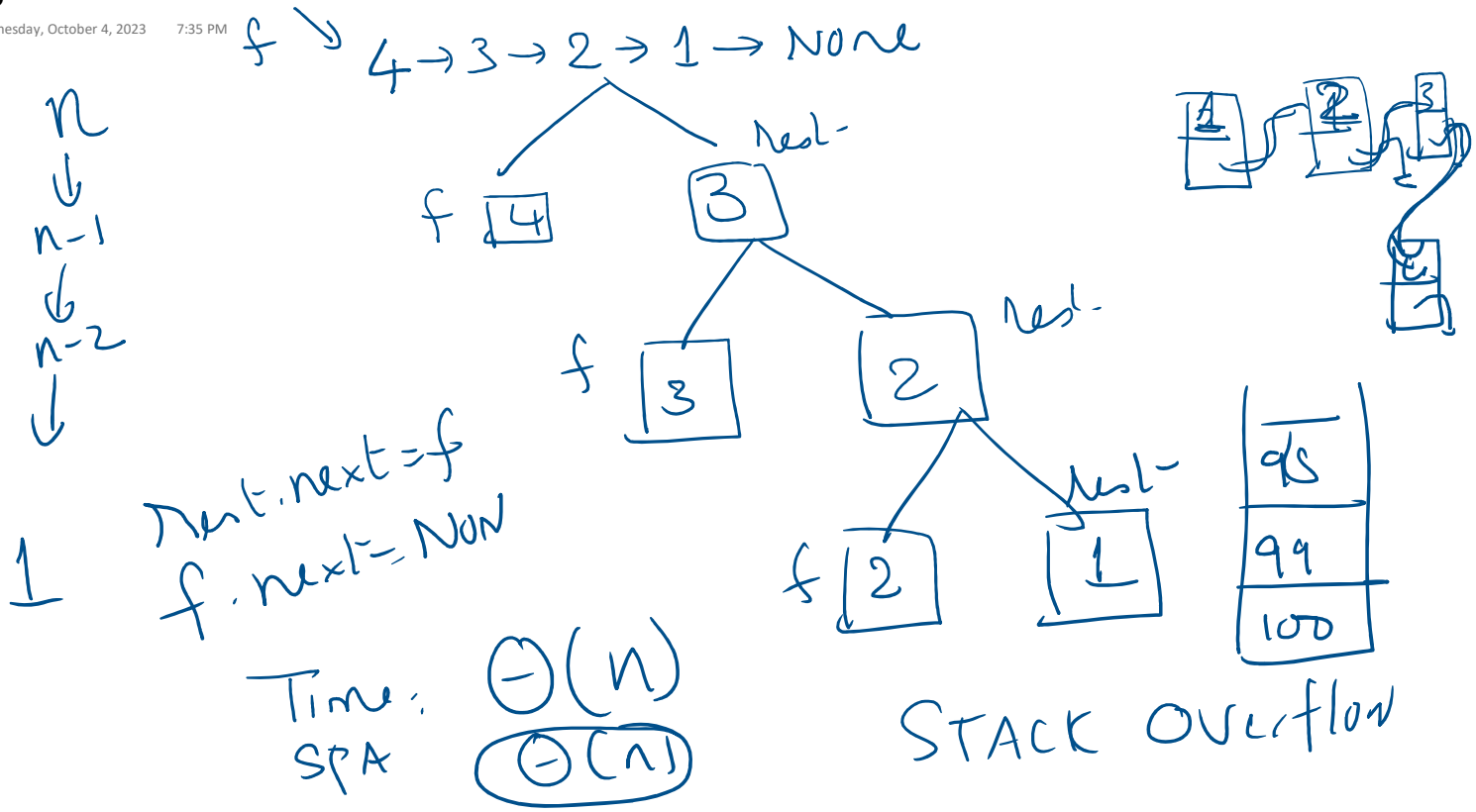




Wednesday, October 4, 2023 7:30 PM



Wednesday, October 4, 2023 7:35 PM

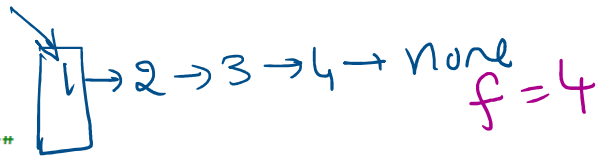


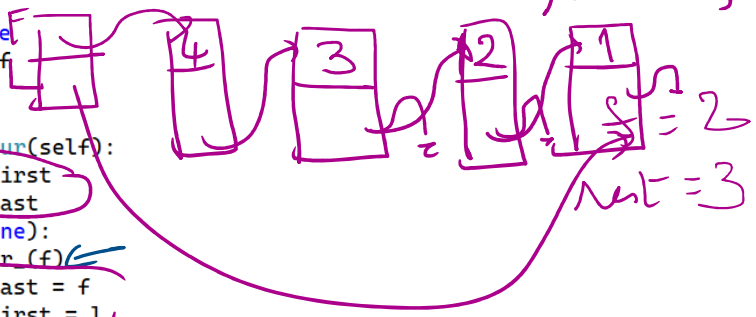
27

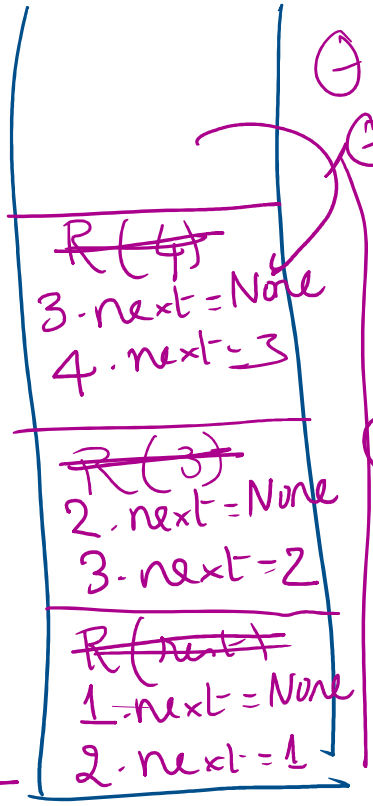
Wednesday, October 4, 2023 7:40 PM

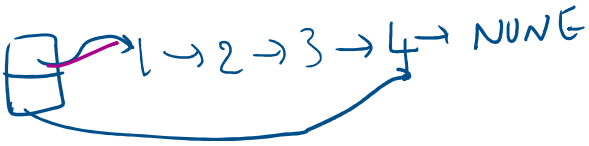
```
################################
def r_r_(self, f:'node'):
    ## Base case
    if (f.next == None):
        return
    ## Divide
    rest = f.next
    self.r_r_(rest);
    ##Conquer
    f.next = None
    rest.next = f


def reverse_recur(self):
    f = self._first
    l = self._last
    if (f != None):
        self.r_r_(f)
        self._last = f
        self._first = l
```



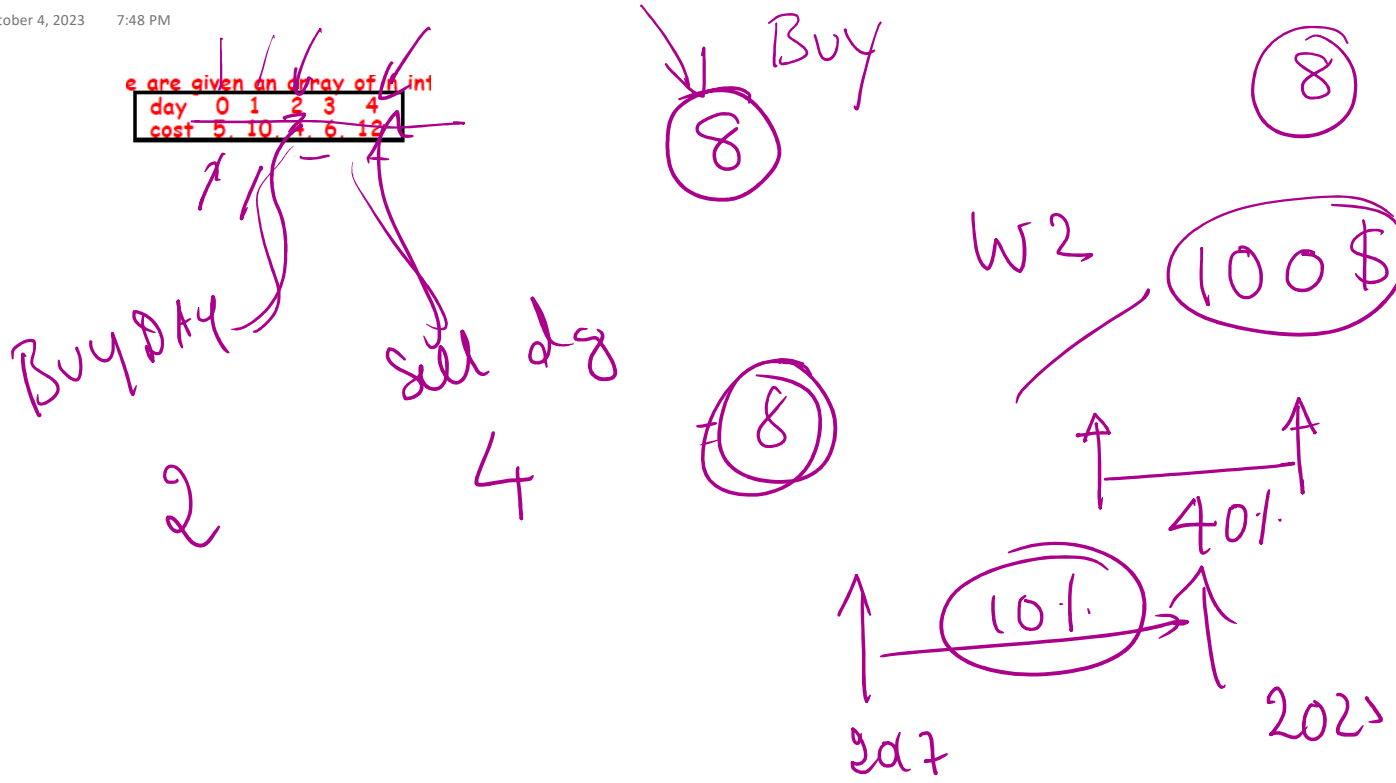






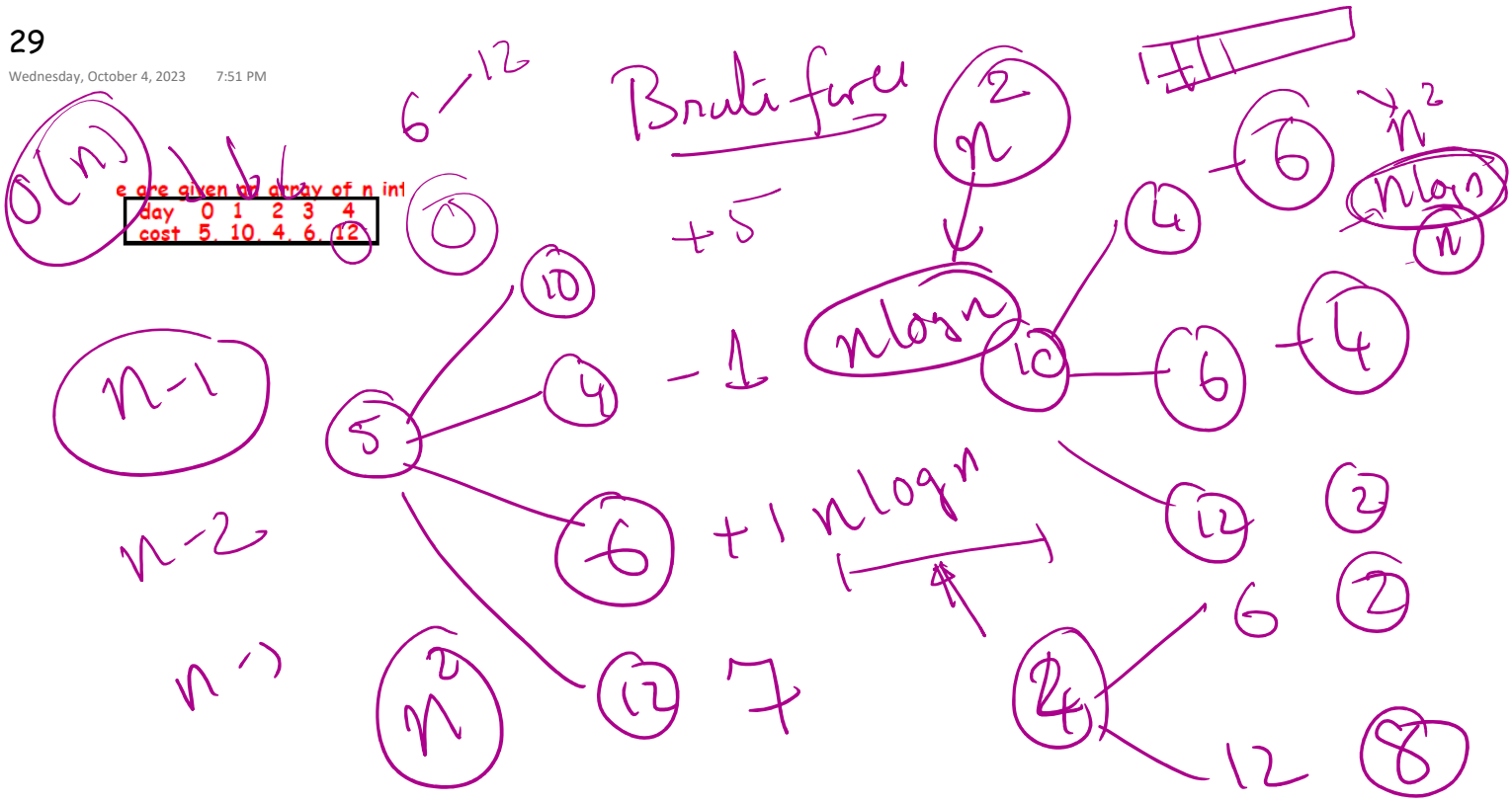
28

Wednesday, October 4, 2023 7:48 PM



29

Wednesday, October 4, 2023 7:51 PM

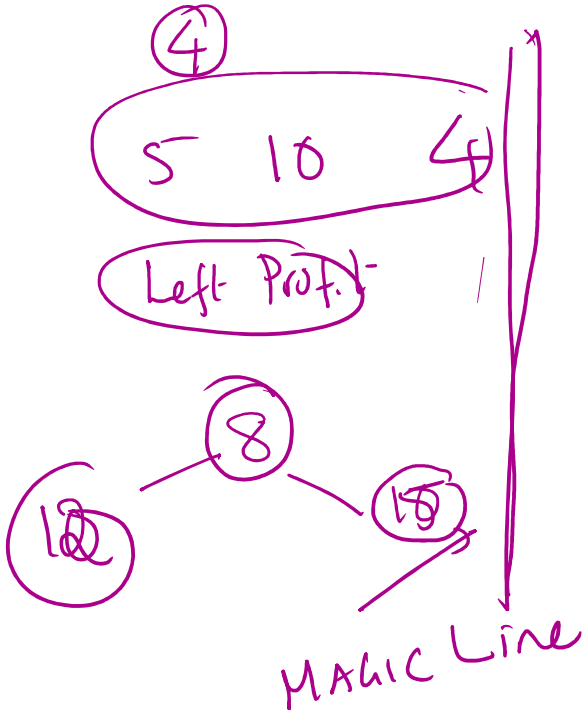


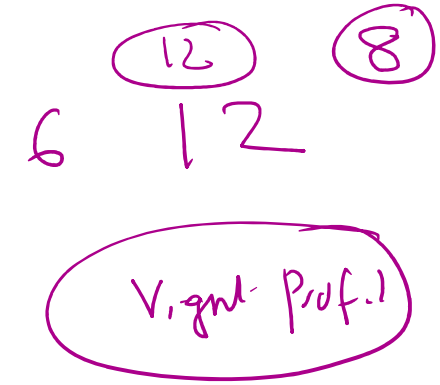
30

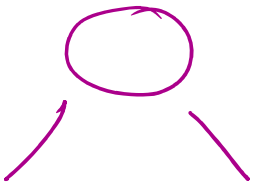
Wednesday, October 4, 2023 7:55 PM

e are given an array of n int

day	0	1	2	3	4
cost	5	10	4	6	12







31

Wednesday, October 4, 2023 7:58 PM

31

e are given an array of n int

day	0	1	2	3	4
cost	5	10	4	6	12