

1.

For find(x) and remove(x) operations, it is easy to show that the amortized cost is  $O(1)$ .

For put(x) operations,

- (i) when doubling does not occur,  
the actual cost of put(x) operation is  $O(1)$
- (ii) when doubling occurs,  
the actual cost is the half of the capacity of the structure prior to the operation is performed.

Thus sum(actual cost of n put(x) operations) is

$$n + \{ \sum(2^{*i}) \mid 1 \leq i \leq k, \text{ where } k = \lceil \log n \rceil \}$$

$$= n + 2 * (2^{*k} - 1) = 2^{k+1} - 2$$

$$\leq 5n - 2 - k$$

$$< 5n$$

Therefore, "5" is enough for the amortized cost of put(x) operation.

2.

S=180 records

n=810 records

m runs

ts = 8 ms

tl = 2 ms

tt = 0.1 ms / record

(a)

(1)  $b = \text{floor}(S / (2k+2)) = \text{floor}(180 / (2k+2))$

(2) time to read a buffer  
 $= (8 + 2 + (1) * 0.1) = (10 + (1) * 0.1) \text{ ms}$

(3) # of buffers per pass  
 $= \text{ceiling}(n / b) = \text{ceiling}(810 / (1))$

(4) input time per pass  
 $= (2) * (3)$

(5) # of passes  
 $= \text{ceiling}(\log m \text{ base } k)$

(6) total input time  
 $= (4) * (5)$

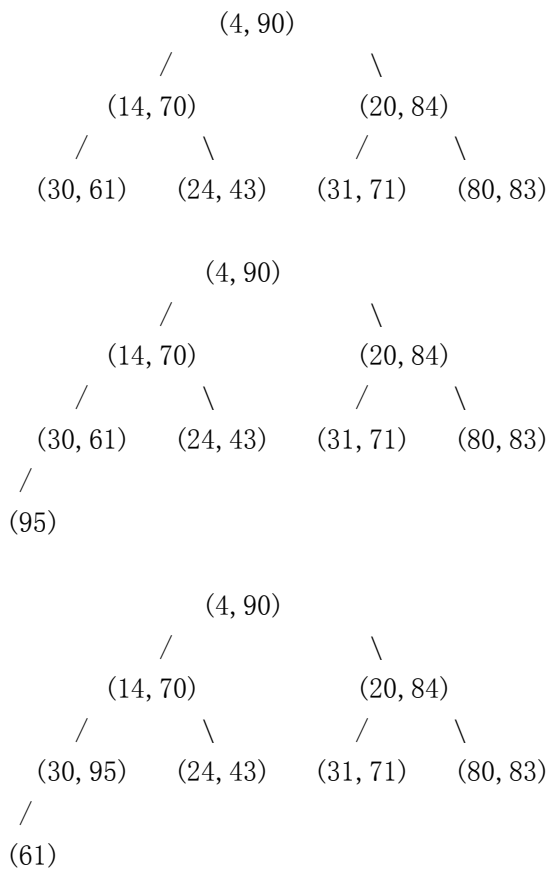
	k=2	k=8
(1)	$180/6 = 30$	$180 / 18 = 10$
(2)	$10+30*0.1 = 13 \text{ ms}$	$10+10*0.1 = 11 \text{ ms}$
(3)	$810/30 = 27$	$810/10 = 81$
(4)	$13*27 = 351 \text{ ms}$	$11*81 = 891 \text{ ms}$
(5)	$\log m \text{ base } 2$	$\log m \text{ base } 8$
(6)	$351 * \log m \text{ base } 2$	$891 * \log m \text{ base } 8$

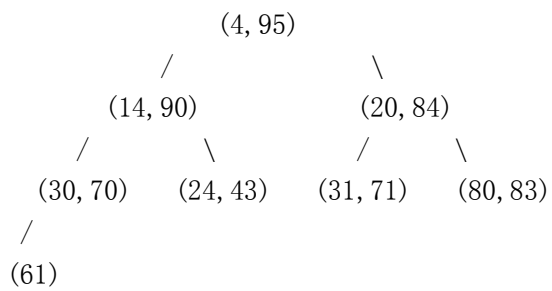
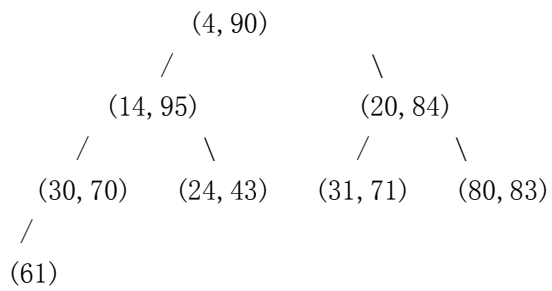
Thus, k=8 is better

(b) Total input time =  $891 * \log m \text{ base } 8$   
 Total output time =  $891 * \log m \text{ base } 8$   
 Total merge time =  $(810*0.1) * \log m \text{ base } 8$

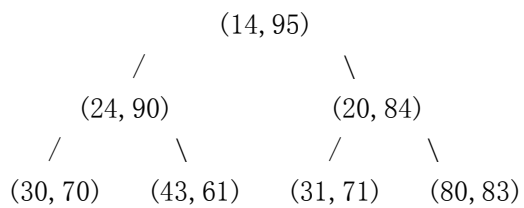
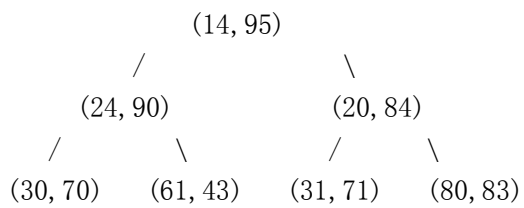
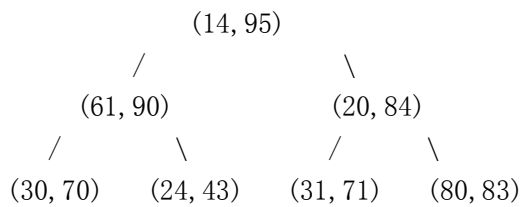
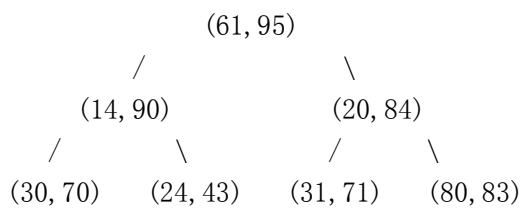
Thus  $(2*891 + 81) * \log m \text{ base } 8 \text{ ms}$

3.



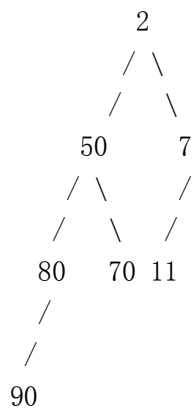


Part (b)



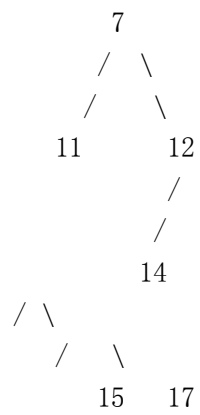
4.

- (a) left tree is already a min leftist tree
- right tree is converted the below tree

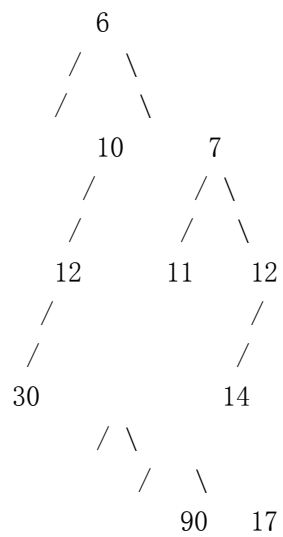


(b)

Step 1.

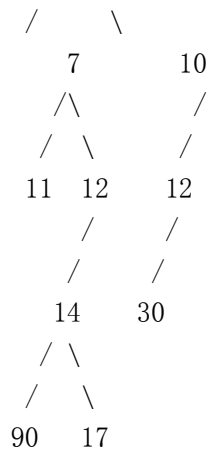


Step2

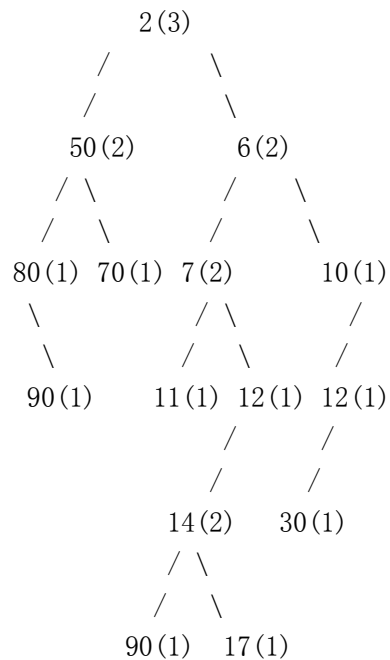


swap the left and right subtree at rooted 6





Step4 : this is a last step of the meld two trees

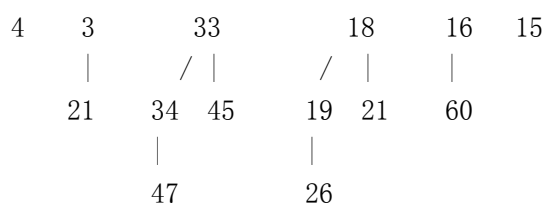


5.

one of the possible Solutions)

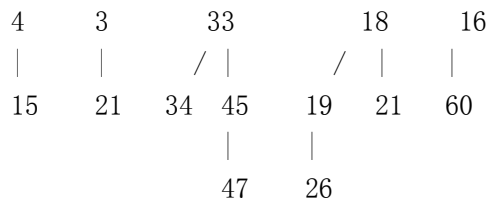
Step1 :

After Delete-Min

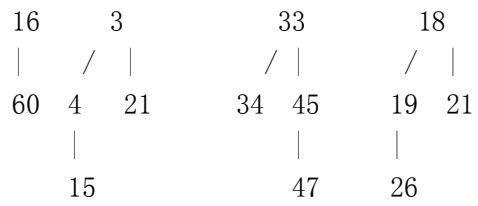


Pairwise combine after delete-Min

(Combine two heaps with degree 0)



(Combine two heaps with degree 1)



(combine two heaps with degree 2)

