def insertanale (rout, heg)  If e-pty tree: -add heg to the tree  else it hey role & root value:  - yo to the loft of the tree  - use recursion  else  i- go to the right of the tree  - use recursion  got holden hopyers)  palance = got holden of root  if (holonce > 1 ne heg > left root value)  left Foot = foft Rotate (feft root)  retern left hother (root)  it (balance < -1 are heg & right root value):  root. right = right Rotate (root)  it (balance < -1 are heg & right root value):  return right public (root)  it (balance > 1 and heg & right root value):  return right public (root)	Write up LABS AVL Tree suraj Render EDGS
else it key value & root value:  go to the loft of the free  - use recorsion  got foot hight = 1 + rax (left of right (holden hory m/s))  palance = gol helme of root  if (halonce > ( ne hey > left root value)  left foot = 1 eft Rotate (teft root)  retoin   left Anthe (root)  if (balonce < - 1 are fig & right root value):  root. right > my hey (teft root value):  root. right nohl (root)	- The same of the
else it key value & root value:  go to the loft of the free  - use recorsion  got foot hight = 1 + rax (left of right (holden hory m/s))  palance = gol helme of root  if (halonce > ( ne hey > left root value)  left foot = 1 eft Rotate (teft root)  retoin   left Anthe (root)  if (balonce < - 1 are fig & right root value):  root. right > my hey (teft root value):  root. right nohl (root)	defined ( rout 400)
else if hey value & root value:  i go to the loft of the free  - Use Micorsion  else  i- go to the right of the tree  - use recorsion  got foot kright = 1 + rax ( left aft right to the horizon)  palance = got helme of root  if (halonce > 1 all hey > left root value)  left foot = 1 eft Notete (test root).  retein left totale (root)  it (falone < -1 are hey & right voot value):  root. right > right kokk (root)  it (balone > 1 ene hey (test root value):  rotun right pakk (root)	, )
else if hey value & root value:  i go to the loft of the free  - Use Micorsion  else  i- go to the right of the tree  - use recorsion  got foot kright = 1 + rax ( left aft right to the horizon)  palance = got helme of root  if (halonce > 1 all hey > left root value)  left foot = 1 eft Notete (test root).  retein left totale (root)  it (falone < -1 are hey & right voot value):  root. right > right kokk (root)  it (balone > 1 ene hey (test root value):  rotun right pakk (root)	If e-phy tree - add has to the troe
else  i-go to the right of the tree  - vec recorsion   get poof height = 1 + max (left aft right  (holden height)  palance = get helice of root  if (halonce > 1 ne heg > left root value)  left voor = A oft Rotete (teft root)  retein   left hoke (root)  if (balance < -1 are ang < right voor value):  root. right = right koke (root)  if (balance < -1 are ang < right voor value):  root. right = right koke (root)	
else  i-go to the right of the tree  - vec recorsion   get poof height = 1 + max (left aft right  (holden height)  palance = get helice of root  if (halonce > 1 ne heg > left root value)  left voor = A oft Rotete (teft root)  retein   left hoke (root)  if (balance < -1 are ang < right voor value):  root. right = right koke (root)  if (balance < -1 are ang < right voor value):  root. right = right koke (root)	else it key value & root value:
else  i-go to the right of the tree  - vec recorsion   get poof height = 1 + max (left aft right  (holden height)  palance = get helice of root  if (halonce > 1 ne heg > left root value)  left voor = A oft Rotete (teft root)  retein   left hoke (root)  if (balance < -1 are ang < right voor value):  root. right = right koke (root)  if (balance < -1 are ang < right voor value):  root. right = right koke (root)	- go to the left of the free
else  i-go to the right of the tree  - vec recorsion   get poof height = 1 + max (left aft right  (holden height)  palance = get helice of root  if (halonce > 1 ne heg > left root value)  left voor = A oft Rotete (teft root)  retein   left hoke (root)  if (balance < -1 are ang < right voor value):  root. right = right koke (root)  if (balance < -1 are ang < right voor value):  root. right = right koke (root)	- Use Mc Corsion
i- go to the right of the tree  - vec recorsion  9st 100t height = 1 + max (left set right  (hildren horizont)  palance = gol hebre of root  if (helonge > 1 ne hey > left root value)  left fort = slott Rotate (teft root)  retein left hotate (root)  if (balance < -1 are hey & right root value):  root. right > right Rotate (root)  if (balance > 1 and hey & right root value):  retein right hotal (root)	
Set 100t. hight = 1 t max (left set right  (hildren horymis)  palance = got holine of root  if (holonge > 1 ne hey > left root value)  left foot = left Rotate (teft root)  reteir n left hobbe (poot)  if (halonge < -1 are hey < right root value):  root. Fight > right kobbe (root)  if (balonge < -1 are hey < tett root value):  root. Fight > right hobbe (root)  if (balonge > 1 and hey < tett root value):  return right hobbe (root)	ele
Set 100t. hight = 1 t max (left set right  (hildren horymis)  palance = got holine of root  if (holonge > 1 ne hey > left root value)  left foot = left Rotate (teft root)  reteir n left hobbe (poot)  if (halonge < -1 are hey < right root value):  root. Fight > right kobbe (root)  if (balonge < -1 are hey < tett root value):  root. Fight > right hobbe (root)  if (balonge > 1 and hey < tett root value):  return right hobbe (root)	i- go to the right of the tree
Set 100t. hight = 1 t max (left set right  (hildren horymis)  palance = got holine of root  if (holonge > 1 ne hey > left root value)  left foot = left Rotate (teft root)  reteir n left hobbe (poot)  if (halonge < -1 are hey < right root value):  root. Fight > right kobbe (root)  if (balonge < -1 are hey < tett root value):  root. Fight > right hobbe (root)  if (balonge > 1 and hey < tett root value):  return right hobbe (root)	- use recorsion
if (holonce > 1 Ne hey > left root value)  left fort = foft Rotete (telt root).  reteir n left totale (root)  if (balonce < -1 are hey & right root value):  root-fight > right kould (root)  it (balonce > 1 one hey (telt root value):  retein right mould (root)	
if (holonce > 1 Ne hey > left root value)  left fort = foft Rotete (telt root).  reteir n left totale (root)  if (balonce < -1 are hey & right root value):  root-fight > right kould (root)  it (balonce > 1 one hey (telt root value):  retein right mould (root)	Stat 100t height = 1 + max ( left as right
if (holonce > 1 Ne hey > left root value)  left fort = foft Rotete (telt root).  reteir n left totale (root)  if (balonce < -1 are hey & right root value):  root-fight > right kould (root)  it (balonce > 1 one hey (telt root value):  retein right mould (root)	Children horyms)
if (holonce > 1 Ne hey > left root value)  left fort = foft Rotete (telt root).  reteir n left totale (root)  if (balonce < -1 are hey & right root value):  root-fight > right kould (root)  it (balonce > 1 one hey (telt root value):  retein right mould (root)	pagace got habase of root
return left hotele (fest root).  return left hotele (poot)  if (palme <-1 are by & right voot volve):  root. 1.3ht = right kotele (voot)  it (balance >1 one hay (tett root value):  return right mobile (root)	
return left hotele (fest root).  return left hotele (poot)  if (palme <-1 are by & right voot volve):  root. 1.3ht = right kotele (voot)  it (balance >1 one hay (tett root value):  return right mobile (root)	it (halonce > 1 we hay > left root value)
return left hothe (poot)  I'f (palme < -1 are high right root value):  riot-1.5ht = right Rober (root)  I'f (balance >1 and high (tett root value):  return right mobile (root)	
return left hothe (poot)  I'f (palme < -1 are high right root value):  riot-1.5ht = right Rober (root)  I'f (balance >1 and high (tett root value):  return right mobile (root)	16th your = Nott Notate (test 100+).
it (balance >1 one hay (tett root value): return right mobile (root)	refurn left hotake (koot)
it (balance >1 one hay (tett root value): return right mobile (root)	
it (balance >1 one hay (tett root value): return right mobile (root)	" ( Palone < - 1 Ars high treat volve);
it (balance >1 one hay (tett root value): return right mobile (root)	right = right kutake (rest)
return right noble (root)	
	and the second of the proof value):
	riting right Noble (root)
retion let + Noble (rus +)	
potein (at thousand)	and heg > Not let, self.
	potein (at thould (rust)
refun pot	rehim aik

EDG3 Rysthatate ( root): PROX JUNE 10 T. Let VOT. Let VOT. y= /1941 100+ trop = left root of 1 reptor y = voot right of root = trap Not. beight = 1 + and ( left and right rout dest g y . hoight = 1 + -1x ( right and 1+11 rot of of vetun j Left Notele (rout): y - root - vyn1 y, let = root vesting = tery Tout . hoight = 1 1 max ( left and right of y. hoish v 11 --- 1 left --- -- 54+ st y) Jehrn y