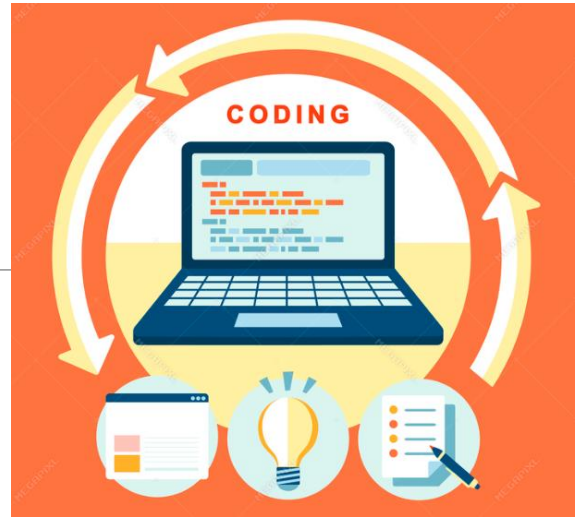
성균관대학교
SUNG KYUN KWAN UNIVERSITY(SKKU)

자 료 구 조

실 습 활 동



균형 탐색 트리

(Balanced Search Tree)

ADT(Abstract Data Type)

AVL Tree ADT

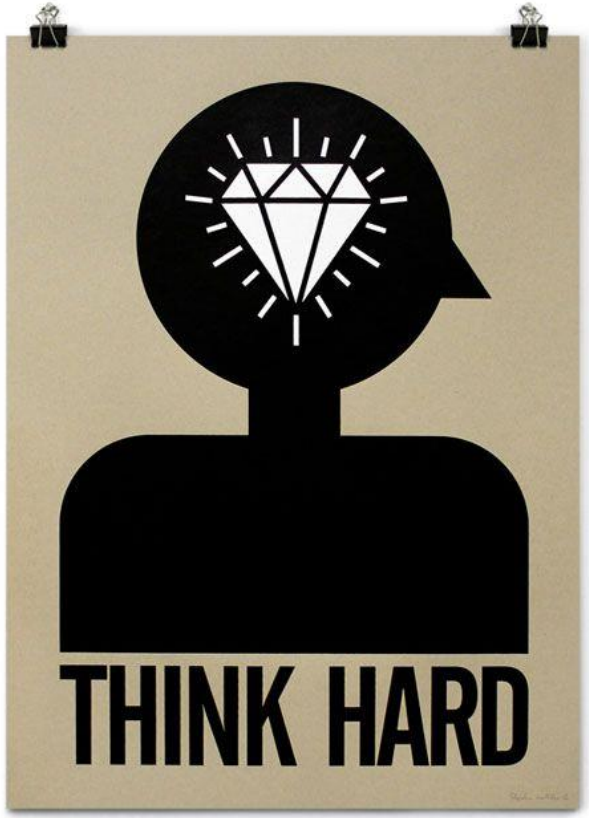
작업: ADT (구현자 관점)	명령어 (사용자 관점)	실행 결과 (자료 관점)
create(my_AVL)	+5	(5)
insert_node(my_AVL, new_node)	+6	(5(,6))
insert_node(my_AVL, new_node) ※ balance_factor(n) where n은 각 노드 ※ left_rotate(my_AVL, 5) 실행 필요	+8	(5(,6(,8)))이 우선 발생 (6(5,8))
insert_node(my_AVL, new_node)	+3	(6(5(3,),8))

작업: ADT (구현자 관점)	명령어 (사용자 관점)	실행 결과 (자료 관점)
insert_node(my_AVL, new_node) ※ balance_factor(n) where n은 각 노드 ※ right_rotate(my_AVL, 5) 실행 필요	+2	(6(5(3(2,),),8)) (6(3(2,5),8))
insert_node(my_AVL, new_node) ※ balance_factor(n) where n은 각 노드 ※ LR_rotate(my_AVL, 6)	+4	(6(3(2,5(4,)),8)) (5(3(2,4),6(,8)))
insert_node(my_AVL, new_node) ※ balance_factor(n) where n은 각 노드 ※ RL_rotate(my_AVL, 6)	+7	(5(3(2,4),6(,8(7,)))) (5(3(2,4),7(6,8)))
print(my_AVL)	P	(5(3(2,4),7(6,8)))
balance_factor(my_AVL, node_value)	B7	0

작업: ADT (구현자 관점)	명령어 (사용자 관점)	실행 결과 (자료 관점)
inorder_traversal(my_AVL)	I	2 3 4 5 6 7 8
right_root_left_traversal(my_AVL)	R	8 7 6 5 4 3 2
get_min(my_AVL)	N	2
get_max(my_AVL)	X	8
find_node(my_AVL, node_value)	F9	Error // Not Exist!
find_node(my_AVL, node_value)	F6	Root-Right-Left

작업: ADT (구현자 관점)	명령어 (사용자 관점)	실행 결과 (자료 관점)
height(my_AVL)	H	3
get_right_child(my_AVL, node)	G3	4
get_left_child(my_AVL, node)	L2	NULL
count_node(my_AVL)	#	7
delete_node(my_AVL, node)	D5	(4(3(2,),7(6,8)))
clear(my_AVL)	C	

자신만의 기능을 3개 추가해보세요!



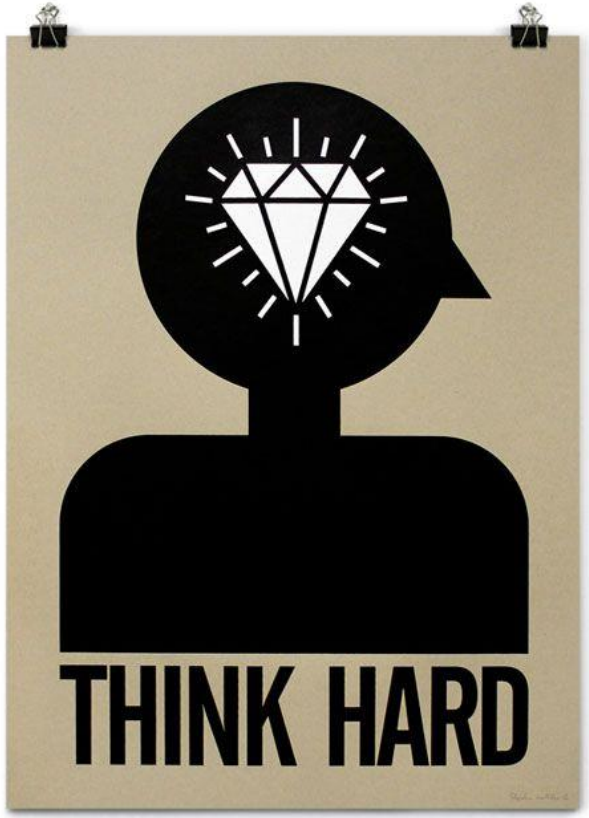
2-3 Tree ADT

작업: ADT (구현자 관점)	명령어 (사용자 관점)	실행 결과 (자료 관점)
create(my_23)	+9	(9)
insert_node(my_23, new_node)	+5	(5,9)
insert_node(my_23, new_node)	+8	(5,8,9) (8((5)()(9)))
insert_node(my_23, new_node)	+3	(8((3,5)()(9)))
insert_node(my_23, new_node)	+2	(8((2,3,5)()(9))) ((3,8)((2)(5)(9)))

작업: ADT (구현자 관점)	명령어 (사용자 관점)	실행 결과 (자료 관점)
insert_node(my_23, new_node)	+4	((3,8)((2)(4,5)(9)))
insert_node(my_23, new_node)	+7	((3,8)((2)(4,5,7)(9))) ((3,5,8)((2)(4)(7)(9))) (5(3((2)()(4))()(8((7)()(9)))))
print(my_23)	P	(5(3((2)()(4))()(8((7)()(9)))))
inorder_traversal(my_23)	I	2 3 4 5 6 7 8 9
right_root_left_traversal(my_23)	R	9 8 7 6 5 4 3 2
get_min(my_23)	N	2
get_max(my_23)	X	9

작업: ADT (구현자 관점)	명령어 (사용자 관점)	실행 결과 (자료 관점)
find_node(my_23, node_value)	F9	Root-Right-Right
height(my_23)	H	3
get_right_child(my_23, node)	G3	4
get_left_child(my_23, node)	L5	3
count_node(my_23)	#	7
delete_node(my_23, node)	D8	((3,5)((2)(4)(7,9))
get_mid_child(my_23, s_node, l_node)	M(3,5)	4
clear(my_23)	C	

자신만의 기능을 3개 추가해보세요!



【도전 프로그램】

- 2-3-4 트리로 입력 자료를 관리하는 프로그램을 구현해보세요.