EL0012: 자료구조 숙제

상명대학교 전기전자컴퓨터학부

숙제 번호	06	점수	/22
제출 날짜	2019년 5월 15일	숙제 디렉토리	~/ds/hw06
학생 이름		학번	

문제 번호	1	점수	8
문제 유형	프로그래밍	프로그램 파일	tree1.c
문제 설명	Template 파일 tree1.h에 정의된 매크로 및 TREE 형을 사용하여 자연수의 binary tree를 array를 사용하여(없는 node 위치에는 -1 저장) 처리하는 아래 함수를 파일 tree1.c에 구현하라. ① void init_tree(TREE * t): tree t를 초기화(모든 node 위치에 -1 저장) 한다. ② void preorder_tree(TREE * t, int root): tree t의 node root에서 preorder로 traverse하면서 각 node의 data를 (%d) 형식으로 출력한다. ③ void inorder_tree(TREE * t, int root): tree t의 node root에서 inorder로 traverse하면서 각 node의 data를 (%d) 형식으로 출력한다. ④ void postorder_tree(TREE * t, int root): tree t의 node root에서 postorder로 traverse하면서 각 node의 data를 (%d) 형식으로 출력한다.		

시험 프로그램 수행 보기	`$ ./test-tree1`
	`===== test-tree1 =====`
	`preorder=(11)(12)(14)(18)(19)(15)(20)(21)(13)(16)(22)(17)`
	`inorder=(18)(14)(19)(12)(20)(15)(21)(11)(22)(16)(13)(17)`
	`postorder=(18)(19)(14)(20)(21)(15)(12)(22)(16)(17)(13)(11)`		

제출물	1.1. 파일 tree1.c에서 학생이 작성한 각 함수를 쓰고 설명하기. (4점) 1.2. 시험 프로그램 test-tree1을 수행한 화면 출력하기. (4점) 1.3. oak 서버의 숙제 디렉토리에 저장된 프로그램 파일.		

문제 번호	2	점수	6
문제 유형	프로그래밍	프로그램 파일	tree2.c
문제 설명	Template 파일 tree2.h에 정의된 매크로 및 TREE 형을 사용하여 자연수의 binary tree를 link를 사용하여 처리하는 아래 함수를 파일 tree2.c에 구현하라. ① NODE *create_node(int item): 하나의 tree node를 malloc을 사용하여 생성하고 item을 저장한 후 그 node의 주소를 return한다. ② void add_child(NODE * root, NODE * left, NODE * right): node root에 node left를 left subtree로 연결하고 node right를 right subtree로 연결한다. ③ int sum_tree(NODE * root): node root가 형성하는 tree의 모든 node에 저장된 자연수를 합한 값을 return한다.		

시험 프로그램 수행 보기	$ ./test-tree2 ===== test-tree2 ===== sum_tree(t0)=660 sum_tree(t1)=420	sum_tree(t2)=240 sum_tree(t4)=230 sum_tree(t5)=160
제출물	2.1. 파일 tree2.c에서 학생이 작성한 각 함수를 쓰고 설명하기. (3점) 2.2. 시험 프로그램 test-tree2를 수행한 화면 출력하기. (3점) 2.3. oak 서버의 숙제 디렉토리에 저장된 프로그램 파일.	

문제 번호	3	점수	8
문제 유형	프로그래밍	프로그램 파일	tree3.c
문제 설명	Template 파일 tree3.h에 정의된 매크로 및 TREE 형을 사용하여 자연수의 key를 가지는 binary search tree를 link를 사용하여 처리하는 아래 함수를 파일 tree3.c에 구현하라. ① NODE *insert_node(NODE * root, int key): node root가 구성하는 binary search tree에 key가 저장될 node를 malloc으로 생성한 후 이 tree의 적절한 위치에 삽입한 후 생성된 새 tree의 root node의 주소를 return한다. 삽입 후 생성된 새 tree는 binary search tree를 구성하여야 한다. ② void print_inorder(NODE * root): node root가 구성하는 binary search tree를 inroder로 traverse하면서 각 node의 key를 (%d) 형식으로 출력한다.		
시험 프로그램 수행 보기	$ ./test-tree3 ===== test-tree3 ===== insert=30 tree=(30) insert=80 tree=(30)(80) insert=90 ... tree=(10)(30)(50)(60)(70)(80)(90) insert=20 tree=(10)(20)(30)(50)(60)(70)(80)(90) insert=40 tree=(10)(20)(30)(40)(50)(60)(70)(80)(90)		
제출물	3.1. 파일 tree3.c에서 학생이 작성한 각 함수를 쓰고 설명하기. (4점) 3.2. 시험 프로그램 test-tree3을 수행한 화면 출력하기. (4점) 3.3. oak 서버의 숙제 디렉토리에 저장된 프로그램 파일.		

주의: 숙제 디렉토리에 학생이 작성한 프로그램 파일(수행되지 않더라도)이 없는 경우 0점 처리함. 숙제 제출 시간 이 후 프로그램 파일을 수정하는 경우 감점 30% 있음.

끝.