

|  |
| --- |
| Binary Tree (배열 구현방식) |
| **과제 5장 보고서(1번 - 배열 구현방식)** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **제 출 일** | **2014. 0** |  | **전 공** | **Business & CSE** |
| **과 목** | **자료구조론** |  | **학 번** | **20101215** |
| **담당교수** |  |  | **이 름** | **정 O O** |

**1. Binary Tree(배열)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Complete Binary Tree의 원리**   |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | |  | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |   **Leftorder의 원리:** 2 \* index  **Rightorder의 원리:** 2\* index + 1  **Index = 0 일 때,**  - 2 \* 0 = 0, …..  - (2 \* 0) + 1 = 1, (2 \* 1) + 1 = 3, ….  **Index = 1일 때,**  - 2 \* 1 = 2, 2 \* 2 = 4, …… , 이런식으로 진행  - (2 \* 1) + 1 = 3, (2 \* 3) + 1 = 7, …… , ……  **증명)**   |  |  |  | | --- | --- | --- | | **Preorder** | **Inorder** | **Postorder** | | template <class T>  void BinaryTree<T>::Preorder(int index){  if (pArr[index] == 0 || index > GetMaxSize())  return;  cout << pArr[index] << " ";  Preorder(2 \* index);  Preorder(2 \* index + 1);  } | template <class T>  void BinaryTree<T>::Inorder(int index){  if (pArr[index] == 0 || index > GetMaxSize())  return;  Inorder(index \* 2);  cout << pArr[index] << " ";  Inorder(index \* 2 + 1);  } | template <class T>  void BinaryTree<T>::Postorder(int index){  if (pArr[index] == 0 || index > GetMaxSize())  return;  Postorder(index \* 2);  Postorder(index \* 2 + 1);  cout << pArr[index] << " ";  } | | **Levelorder** | | | | template <class T>  void BinaryTree<T>::Levelorder(int index){  if (pArr[index] == 0 || index > GetMaxSize())  return;  cout << pArr[index] << " ";  Levelorder(index + 1);  } | | |   **자료구조론 교제에 있는 원리를 적용해본 결과** |

**2. 코드 결과**

|  |
| --- |
| #include <iostream>  #include "equiv.h"  using namespace std;  void Equivalence(char\*);  // 인자 존재 여부(파일 이름 파악)  bool IsArgv(int argc, char\*\* argv){  int i = 0;  while (i < argc)  {  if (i > 1)  break;  i++;  }  if (i == 2)  return true;  else  return false;  }  char\* GetFileName(char\*\* argv){  return argv[1];  }  int main(int argc, char\*\* argv)  {  if (IsArgv(argc, argv) == true)  Equivalence(GetFileName(argv));  else  cout << "인자가 준비되어 있지 않습니다." << endl;    } |

Figure 1) Main.cpp