Trường Đại học Khoa Học Tự Nhiên

Khoa Điện tử - Viễn thông

**THỰC HÀNH CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT**

**HÌNH THỨC 2**

Họ và Tên: Nguyễn Ngọc Nguyên

MSSV: 20200058

Đề bài : Cài đặt cây nhị phân tính giá trị 1 biểu thức

**Cài đặt:**

|  |
| --- |
| #include<iostream>  #include<string>  using namespace std;  typedef struct NODE  {      string data;      NODE\* left;      NODE\* right;  };  typedef struct NODE\* TREE;  void khoiTao(TREE &root)  {      root = NULL;  }  NODE\* newNode(string data)  {      NODE\* node = new NODE;      node->data = data;      node->left = NULL;      node->right = NULL;      return node;  } |

* Đầu tiên em thêm thư viện iostream và string để chứa chuỗi dữ liệu
* Tiếp theo định nghĩa một kiểu dữ liệu cấu trúc NODE chứa dữ liệu (string), và hai con trỏ ‘left’, ‘right’ để trỏ tới nút trái và phải của nút hiện tại.
* Tiếp theo định nghĩa dữ liệu TREE dùng con trỏ trỏ tới kiểu NODE
* Hàm ‘khoiTao’ để khởi tạo giá trị NULL cho cây nhị phân
* Hàm ‘newNode’ dùng để tạo một nút mới cho cây nhị phân với kiểu dữ liệu ‘data’

|  |
| --- |
| bool checkNode(NODE \*node)  {      return node->left == NULL && node->right == NULL;  }  float tinhToan(string data, float x, float y)  {      if (data == "+"){          return x + y;      }      if (data == "-") {          return x - y;      }      if (data == "\*") {          return x \* y;      }      if (data == "/") {          return x / y;      }      return 0;  }  float NLR(TREE root)  {      if (root == NULL){          return 0;      }      if (checkNode(root)){          return stod(root->data);      }      float a = NLR(root->left);      float b = NLR(root->right);      return tinhToan(root->data, a, b);  } |

* Tại đây em định nghĩa hàm ‘checkNode’ nhận vào một con trỏ trỏ tới struct NODE và trả về một giá trị boolean.
* Hàm sẽ kiểm tra xem con trỏ ‘left’ và ‘right’ của nút đó có trỏ tới NULL hay không (tức nút không có nút con). Nếu cả hai con trỏ đề trỏ đến NULL (nút lá) thì trả về ‘true’. Ngược lại trả về ‘false’.
* Hàm ‘tinhToan’ nhận 3 đối số là data(string) đại diện cho toán tử toán học (‘ + ’ , ‘ - ’ , ‘ \* ’ hoặc ‘ / ’), và hai giá trị “x”, “y” kiểu float đại diện cho toán hạng. Hàm sử dụng các giá trị này để thực hiện phép toán tương ứng và trả về kết quả. Nếu “data” không phải toán tử hợp lệ. chương trình trả về giá trị 0.
* Hàm “float NLR(TREE root)” đệ quy theo thứ tự trước của cây biểu thức nhị phân bắt đầu từ nút gốc. Hàm kiểm tra xem nút hiện tại có phải là nút lá không bằng cách gọi đến hàm “checkNode()”.
* Nếu nút hiện tại là nút lá, chuyển đối dữ liệu lưu trữ trong nút từ chuỗi sang số bằng cách sử dụng hàm ‘stod()’ và trả về.
* Nếu nút hiện tại không phải nút lá. Đệ quy tính toán cây con bên trái và cây con bên phải gọi lại hàm “NLR”, lấy kết quả của toán hạng và truyền cho hàm “tinhToan” để thực hiện phép tính. Toán tử được lưu trữ trong nút hiện tại. sau đó trả về kết quả của phép tính.

|  |
| --- |
| int main()  {      TREE root ;      khoiTao(root);      root = newNode("-");      root->left = newNode("/");      root->right = newNode("+");      root->left->left = newNode("\*");      root->left->right = newNode("+");      root->right->left = newNode("+");      root->right->right = newNode("/");      root->left->left->left = newNode("+");      root->left->left->right = newNode("4");      root->left->right->left = newNode("-");      root->left->right->right = newNode("2");      root->right->left->left = newNode("\*");      root->right->left->right = newNode("2");      root->right->right->left = newNode("8");      root->right->right->right = newNode("4");      root->left->left->left->left = newNode("/");      root->left->left->left->right = newNode("3");      root->left->right->left->left = newNode("7");      root->left->right->left->left = newNode("7");      root->left->right->left->right = newNode("5");      root->right->left->left->left = newNode("4");      root->right->left->left->right = newNode("/");      root->left->left->left->left->left = newNode("3");      root->left->left->left->left->right = newNode("1");      root->right->left->left->right->left = newNode("6");      root->right->left->left->right->right = newNode("1");      cout << "ket qua: " << NLR(root) << endl;      return 0;  } |

* Khai báo biến root thuộc kiểu dữ liệu TREE, là một con trỏ trỏ tới nút gốc của cây
* Gọi hàm “khoiTao” để tạo cây rỗng
* Tạo nút gốc của cây và đặt giá trị là (-)
* Tạo các nút con trực tiếp từ nút gốc theo như hình dưới đây:

A picture containing device

Description automatically generated

Biểu thức (((((3/1)+3)\*4)/((7-5)+2))-(((4\*(6/1))+2)+(8/4))) = -22

* Gọi hàm NLR duyệt và tính toán giá trị biểu thức từ cây.
* Xuất giá trị.
* Kết thúc chương trình.

**Hình chụp kết quả chương trình:**

Text

Description automatically generated