提出日　令和2年08月19日

アルゴリズムとデータ構造　第回課題レポート

学籍番号（A19117）
氏名（永尾優磨）

【課題】

1.　木構造について、まとめよ。

枝分かれして広がり、一つの親に対して複数の子を持つ構造のこと

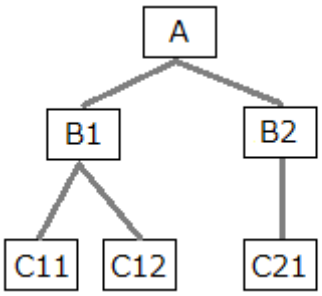

図1.1　木構造の模式図

Aを起点として、下に行くほど広がっている。この図1のような構造が木構造だ。図1に押して資格で表現した要素を「ノード」といい、要素同士繋ぐ線の部分を「エッジ」と言う。

根っこの方（上）が「親」であり、広がっていく先が、「子」となる。

図1.1内のAとB1の関係を言葉で表すと、AはB1にとっての親であり、B1はAにとっての子である。この言葉を図にすると図1.2のようになる。

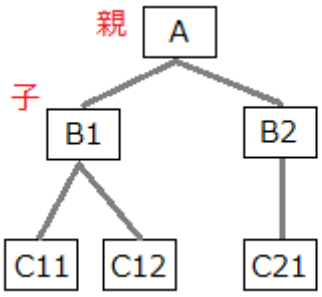

図1.2 親子の関係

2.　適当な木構造ので-他を作り、データ投入、検索、表示を行ってみよ。

【作成したプログラム】

```
public class Node {

    public static int count = 0;

    private String text;
    private Node left;
```

```
    private Node right;
    private int id;

    Node(String text) {
        setText(text);
        setLeft(null);
        setRight(null);
        setId(count++);

    }

    public String getText() {
        return text;
    }

    public void setText(String text) {
        this.text = text;
    }

    public Node getLeft() {
        return left;
    }

    public void setLeft(Node left) {
        this.left = left;
    }

    public Node getRight() {
        return right;
    }

    public void setRight(Node right) {
        this.right = right;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

}
```

プログラム1　Node

```
public class Tree {
```

```
    private Node root;
    private static int count;

    Tree(String text) {
        root = new Node(text);
        count = 1;

    }

    public void add(String text) {
        Node t = new Node(text);
        Node p = root;
        Node old = null;

        while (p != null) {
            old = p;
            if (p.getText().compareTo(text) > 0) {
                p = p.getLeft();
            } else {
                p = p.getRight();
            }
        }
        if (old.getText().compareTo(text) > 0) {
            old.setLeft(t);
        } else {
            old.setRight(t);

        }
        count++;
    }

    public Node search(String text) {
        Node p = root;
        while (p != null) {
            if (p.getText().equals(text) == true) {
                System.out.printf("見つかりました。%s id=%d¥n", p.getText(), p.getId());

                break;
            } else if (p.getText().compareTo(text) > 0) {
                p = p.getLeft();
            } else {
                p = p.getRight();
            }
        }
        if (p == null) {
            System.out.println("見つかりません。");

        }
```

```
        return p;
    }

    public void print(Node p) {
        int l, r;
        if (p != null) {
            print(p.getLeft());

            if (p.getLeft() != null) {
                l = p.getLeft().getId();
            } else {
                l = -1;
            }

            if (p.getRight() != null) {
                r = p.getRight().getId();
            } else {
                r = -1;
            }
            System.out.printf(" %4d --- %4d)%10s --- %4d¥n", l, p.getId(), p.getText(), r);

            print(p.getRight());

        }
    }

    public int getCount() {
        return count;
    }

    public Node getRoot() {
        return root;
    }

}
```

プログラム2　Tree

```
public class TreeApp {
    public static void main(String[] args) {
        String[] data = { "Ymamot", "Takahashi", "Suzukjkdfpakjdpfai", "Tanaka", "Nakamura", "Watana
be", "Katoh",
                "Satoh", "Itoh", "helsinki", "kobayashi" };
        Tree t = new Tree(data[0]);
        for (int i = 1; i < data.length; i++) {
```

```
            t.add(data[i]);
        }
        t.print(t.getRoot());

        t.search("Tanaka");
        t.search("text");

    }
}
```

プログラム3　TreeApp

【プログラムの解説】
授業スライドをほぼそのまま写した。改善した点として、serach関数を変更した。その結果、TreeApp内からif文をなくすことに成功し、見やすくなった。この改善をしたことで、Java上で、nullを使ったif文を書くことに成功した。さらなる改善点としては、TreeAppないに入れるデータを外部のテキストファイルなどから読み込めるようにすることが考えられる。

【結果】

```
   -1 ---    8)      Itoh ---   -1
    8 ---    6)     Katoh ---   -1
    6 ---    4)  Nakamura ---    7
   -1 ---    7)     Satoh ---   -1
    4 ---    2)    Suzuki ---   -1
    2 ---    1) Takahashi ---    3
   -1 ---    3)    Tanaka ---    5
   -1 ---    5)  Watanabe ---   -1
    1 ---    0)    Ymamot ---    9
   -1 ---    9) kobayashi ---   -1
見つかりました。Tanaka id=3
見つかりません。
```

図1　TreeAppの出力結果

【考察】
木構造について理解を深めることができた。
木構造を一言で言うと、一つの点から複数の点が枝分かれする構造と言える。
この構造は、色々な組織に当てはめることができる。例えば、会社がある。会社は、社長を一番上においた木構造である。社長の下に複数の役員や事業があるということを考えると、木構造といえる。

【参考文献】

1. 「アルゴリズムとデータ解析の授業スライド」
2. 増井敏克(2020)「Pythonで始めるアルゴリズム入門」翔泳社
3. 「木構造とは｜「分かりそう」で「分からない」でも「分かった」気になれるIT用語辞典」〈https://wa3.i-3-i.info/word14397.html〉
4. 「参考文献の書き方」〈http://www7a.biglobe.ne.jp/nifongo/ron/ron_04.html〉