```
/*
 *  OutlineModule.h
 *
 *
 *  Created: 3/24/2017 by Ryan Tedeschi
 */

#ifndef OUTLINEMODULE_H
#define OUTLINEMODULE_H

#include <string>
#include <iostream>
#include <vector>
#include "../../shared/CASP_Plugin/CASP_Plugin.h"
#include "../../shared/Printable/Printable.h"

using namespace std;

enum EntryType { Start, MethodCall, Process, Loop, Decision, EndDecision, IO, End };
class OutlineModule;
class Outline;
class Node;
class Edge;

class OutlineModule : public CASP_Plugin {
    public:
        OutlineModule();
        virtual CASP_Return* Execute(Markup* markup, LanguageDescriptorObject* source_ldo, vector<arg> fnArgs, CASP_Return* inputReturn = NULL);

    private:
        vector<Outline*> GetAllOutlines(Markup*);
        Outline* GetRootOutline(vector<Markup*>);
        Outline* GetFunctionOutline(Markup*);
        CASP_Return* FormatData(vector<Outline*>);

        Node* stripProcess(Markup*, Outline*, Node*, string = "");
        Node* stripMethodCall(Markup*, Outline*, Node*, string = "");
        Node* stripDecision(Markup*, Outline*, Node*, string = "");
        Node* stripFor(Markup*, Outline*, Node*, string = "");
        Node* stripWhile(Markup*, Outline*, Node*, string = "");
        Node* processStatement(Markup*, Outline*, Node*, string = "");
        Node* processBlock(Markup*, Outline*, Node*, string = "");
};

class Outline : public Printable {
    public:
        Outline();

        Node* AppendBlock(EntryType, string, Node*);
        Node* AppendBlock(EntryType, string, Node*, string);
        Node* AppendBlock(Node*);
        void Print();

        GenericArray* Output();

    private:
        vector<Node*> nodes;
        int maxId = 0;
        Node* head = NULL;
};

class Node : public Printable {
    public:
        Node(string, EntryType, int);

        Edge* AddEdgeTo(Node*);
```

```
67          Edge* AddEdgeTo(Node*, string);
68          Edge* AddEdgeFrom(Node*);
69          Edge* AddEdgeFrom(Node*, string);
70          void Print();
71
72          GenericObject* Output();
73
74          int id;
75          string data;
76          EntryType type;
77
78      private:
79          vector<Edge*> edges;
80
81  };
82
83  class Edge : public Printable {
84      public:
85          Edge(Node*, Node*, string);
86          Edge(Node*, Node*);
87          void Print();
88
89          GenericObject* Output();
90
91          string data = "";
92          Node* source = NULL;
93          Node* target = NULL;
94
95      private:
96  };
97
98  #endif
```