

Homework 3: Pretty Printing

Instructor : Kiayias

Out Date:

10/11/2007

Due Date:

10/23/2007

Objectives

The purpose of this homework is to learn how to produce an abstract syntax tree as a side-effect of the parsing process. At the same time you will get your first “translation” of the source C--program : your parser will take advantage of the generated abstract syntax tree to “pretty print” the source program. Each node of the abstract syntax tree will be an instance of a class `ASTnode` that will be extended in various ways to facilitate the various components of a C--program. Each extension of the basic node class will also provide an implementation of a method `traverse` that will facilitate the pretty printing.

Handout

To obtain your assignment, go to the class website in the handouts section and download `hw3-package.zip`. Similar to previous homeworks, this archive is a template project that you can import into Eclipse (Use Import -> Existing Projects into Workspace). It contains support code and basic structure. It is up to you to “fill in” the blanks. For your convenience I have included the C--grammar but feel free to toss it and use your own from homework 2. The grammar I give you has no error productions; include your own error productions and error handling that you have done in homework 2. The project now includes a new package called `ast` that contains a number of classes all of which extend the following abstract class:

```
package ast;

public abstract class ASTnode {
    public int line_number;
    public int column_number;
    public Integer nodevalue;
    public String nodelabel;
    public abstract String toString();
    public abstract String traverse(int depth);
    public String tabulator(int depth)
    {
        String my_string = "";
        for (int j=0;j<depth;j++) my_string += "\t";
        return my_string;
    }
}
```

The abstract syntax tree is built by inserting the appropriate actions within the `parser.cup` file that construct the various nodes of the tree. Some of the actions are provided for your reference but in most cases you will have to fill in the missing information. Inside the package `ast` you will find all other classes. Nevertheless in most classes the instantiation of the method `traverse` is missing; you will have to provide this method.

In the zip file you will also find a file called `test.cmm` written in C--. Make sure that your parser successfully pretty prints the given source.

Remark. Feel free to toss away the grammar I give you and substitute it for the one you have built from homework #2.

Pretty Printing

To demonstrate by example what is required in this project consider the following syntactically correct C--program :

```
class a_stupid_class{ int
method1      ( int
n
){return n+1;} int method2(int x)
{ int i; for (i=1;
           i<10;i++)
{ i=i*2; } return x+i;
}};
```

It is hard to read as you see. Your pretty printer should transform it to the following (much easier to read) program:

```
class a_stupid_class{
    int method1(int n) {
        return (n+1);
    }
    int method2(int x) {
        int i;
        for (i=1;(i<10);i++)
            {
                i=(i*2);
            }
        return (x+i);
    }
};
```

Remark. Prettiness of programs is in the eye of the beholder to some extent, so very *tiny* deviations from the above would be tolerable from the grading point of view.

Hand-in

Your solution should be an archive of the whole project named **hw3-LASTNAME.zip** (that should compile correctly and pretty print) and should include the following:

1. The file **parser.cup** that contains the grammar implemented in JavaCUP with the appropriate actions completed.
2. All the classes of the package **ast** appropriately completed.

You submit your homework by midnight on the due date as a zip archive exported from eclipse. The e-mail address to submit is c244fa07@cse.uconn.edu.

Have fun!