Project 2: Type checking and Intermediate Code Generation

Due Nov 11, 2010 (corrections may be made until final)

In this project, you are required to apply type checking to the C code parsed by your project1 and to generate an intermediate representation of the type-checked code. Your IR could be either a high-level AST (abstract syntax tree) or a low-level three address code. But each expression in the IR must include a type as part of its representation. After parsing an input code, e.g., the following sample input, your project needs to print out the typed IR (AST or three-address code) to standard output or external files as result.

```
float a[100][100], b[100][100], c[100][100];
int i, j, k;
i = 0;
while (i < 100) {
  j = 0;
  while (j < 100) {
    if (!(c[i][j] == 0))
       c[i][j] = 0;
    k = 0;
    while (k < 100) {
       c[i][j] = c[i][j] + a[i][k] * b[k][j];
       k = k + 1;
    }
    j = j + 1;
  }
  i = i + 1;
}
```

Depending what languages you chose to implement your project1, you may continue implementing project2 using C/C++/Java, POET, or any other mainstream programming language. Please document clearly your choice of languages as well as how to compile and run your program. Make sure that you make very clear which approach you have chosen to implement the project, especially if you are using an approach different from any of the options listed in this assignment. If you used any tool beyond those mentioned in this assignment, you also need to make sure that we have access to the same tool (letting us know how to download it). And if you have chosen a language different than C, you must provide your own test file which is equivalent to the one given.

Submit your project (combination of project1 and project2) by packaging up your directory into a gzip file and then submitting it online at

`http://www.cs.utsa.edu/~cs5363`

Suppose your entire project is in a directory named “project2”. Goto the parent directory of *project2* and type

```
> tar -cf project2.tar project2
> gzip project2.tar
```

A file *project2.tar.gz* will be generated. Test your package by copying it to another directory, say *tmp*, and then type

```
> tar -zxf project2.tar.gz
```

The package should unpack to a directory that contains your submission. After validating the package, submit it by uploading it online.