Assignment 1 by Karan Patel and Sergey Korobkov:

Managing processing workload between the scanner (tokenization) and parser (grammar) was an important challenge for us. We aimed to determine how tokens (FLEX) should be created to accurately obtain 100% coverage of the MiniGLSL language while minimizing code complexity in grammar definition (BISON)

For example when scanning characters such as "{" , "(" and "]" , we had to decide if we should match individual braces separately (i.e. "{" and "}" are enumerated as unique tokens) or match them identically (i.e. "{" and "}" are the same enumerated token).

Rules to capture scoping parenthesis:
```
\{      {
        yTRACE(L_SCOPE);
        return L_SCOPE;
}

\}      {
        yTRACE(R_SCOPE);
        return R_SCOPE;
}
```

The former option would result in more code duplication in the scanner, but it would simplify token parsing in the grammar definition phase. The latter option would logically have the opposite effect.

Acknowledging that a concise grammar definition would greatly simplify the development of future dependant components, we opted in favour of matching individual braces separately. Similarly, for other rules in the parser, we choose to break our tokens into as many logical fragments as possible.

This strategy is clear when applied to rules recognizing vector data types:

```
"vec"[234]  {
        char* temp = yytext+3;
        yylval.num = atoi(temp);
        yTRACE(T_VEC);
        return T_VEC;
}
"bvec"[234]  {
        char* temp = yytext+4;
        yylval.num = atoi(temp);
        yTRACE(T_BVEC);
```

```
      return T_BVEC;
}
"ivec"[234]  {
      char* temp = yytext+3;
      yylval.num = atoi(temp);
      yTRACE(T_IVEC);
      return T_IVEC;
}
```

Furthermore we have also recorded a value associated with certain tokens to avoid ambiguity. "Float" and "Int" tokens have the coressponding values associated with them. The token that returns an operand has an ASCII value associated with it to help Bison identify which operand it is. Identifiers have strings associated with them to help Bison distinguish between variable identifiers (for example).

We have made the above decisions to make sure that the process of creating the grammar rules will be easier.