program:

type program=((var_type * string) list) * (fdecl list)

program 是由两个 List 组成的 tuple，在 ocaml 中表示为([],[]).

- 第一个 list 是由变量的声明（var_type * string）组成的 list，比如 int a; ellipse e1;
- 第二个 list 是由函数的声明组成的 list， 比如：

```
def int func(int a, int b)
{
  return a+b;
}
```

fdecl:

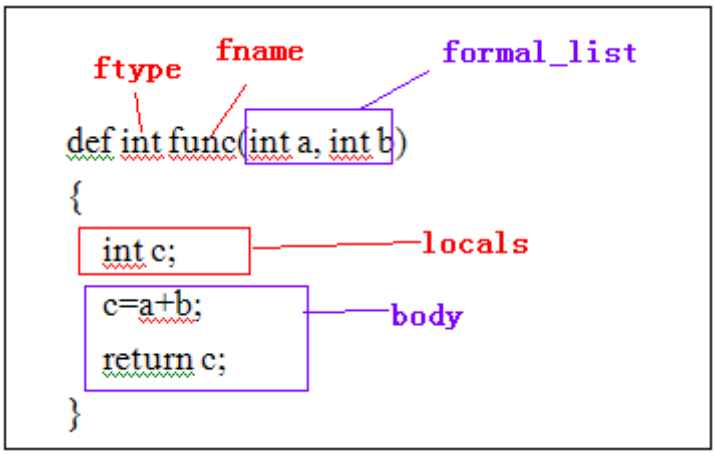
```
fdecl=
    {
        ftype:var_type;
        fname:string;
        formal_list: (var_type * string) list;
        locals: (var_type * string) list;
        body: stmt list

    }
```

fdecl 是 record 类型, 这个类型中有五个元素：

- ftype 元素：类型为 var_type (var_type 是 enum 类型，穷举 ALG 中的所有数据类型， 比如 string， int，point 等)；var_type 的定义如下：

```
type var_type=
| INT_TYPE
|FLOAT
|ARRAY
|STRING
|VOID
|BOOLEAN
| POINT
|LINE
|POLYGON
|ELLIPSE
```

- fname 元素: 函数名，类型为 string；
- formal_list：类型为由（变量类型，变量名）这样的 tuple 组成的 list。formal_list 代表的是形参列表；
- locals：类型为由（变量类型，变量名）这样的 tuple 组成的 list。locals 代表的是函数内部的局部变量定义（在这种定义中，所以局部变量都必须出现在最前面）。
- body：类型为由 stmt 组成的 list。



Stmt:

stmt=

| ExStmt of expr
| Return of expr
| Block of stmt list
| If of expr * stmt * stmt
| For of expr * expr * expr * stmt
| While of expr * stmt

- ExStmt(expr)：表示“expr;”形式的 statement
- Return of (expr): 表示“return expr;”形式的 statement
- Block of stmt list: 表示 “{stmt1 stmt2…stmtN}”形式的 statement
- If of expr * stmt * stmt: 表示的是“if(expr) stmt”或者“if(expr) stmt else stmt”，在第一种形式中，第二个 stmt 为空
- For of expr * expr * expr * stmt：表示 for loop，前三个 expr 分别代表 for loop 中的三个表达式，stmt 代表 for loop 的 body
- While of expr * stmt：表示 while 循环

expr:

type expr=

| NUM of float
| INT of int
| ID of string
| Binop of expr * op * expr
| Assign of string * expr
| Call of string * expr list
| String of string
| PointEx of point
| LineEx of line
| PolygonEx of polygon
| EllipseEx of ellipse

- NUM of float: float 类型常数
- INT of int：int 类型常数
- ID of string: 标识符 identifier
- Binop of expr * op * expr：两个表达式的运算结果
- Assign of string * expr：赋值表达式（string 代表标示符，将 expr 赋值给标示符）
- Call of string * expr list：函数调用，string 代表函数名，expr list 代表函数的实参
- String of string：string 类型常量
- PointEx of point：point 类型常量或者变量（point 的形式可以是[expr,expr]，由于 expr 中可以有 identifier，所以这里不完全是常量，但是这里的变量的含义并不是通常意义上的变量的含义）
- LineEx of line: line 类型常量或者变量
- PolygonEx of polygon：polygon 类型常量或者变量
- EllipseEx of ellipse：ellipse 类型常量或者变量

Point

point=

| Point of expr * expr

Point 的形式可以有多种，便于理解，举例如下：

- [1,2]
- [a,b]
- [1+2,a*b]

以上的例子是指 parser 可以识别的模式，ast 中是两个表达式组成的 tuple（*连接的元素是共同组成 tuple 的元素）

Line:

line=

| Line of point * point

Line 由两个 point 组成的 tuple，parser 可以识别的模式举例如下：

- [[1,2],[a,b]]
- [[1+2,a*b],[a,2]]
-

Polygon:

polygon=

| Polygon of point list

Polygon 的 AST 模型是由 point 组成的 list，parser 可以识别的模式举例如下(三个点以上)：

- [[1+2,a*b],[a,2]，[1,2]]
- [[1+2,a*b],[a,2]，[1,2],[q,a+b]]

Ellipse

ellipse=

| Ellipse of point * expr * expr

Ellipse 由一个 point 和两个 expr 组成的 tuple，point 代表圆心，两个 expr 分别代表长短轴的长度。Parser 可以识别的模式举例如下：

- [[1,2],1+a,a*b]