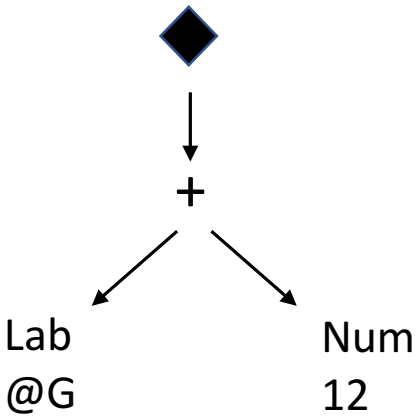
Tree pattern matching worksheet.

Algorithm:

```
Tile(n)
  Label(n) ← existing label
  if n is a binary node then
    Tile(left(n))
    Tile(right(n))
    for each rule r that matches n's operation
      if left(r) ∈ Label(left(n)) and right(r) ∈ Label(right(n))
        then Label(n) ← Label(n) ∪ {r}
  else if n is a unary node then
    Tile(left(n))
    for each rule r that matches n's operation
      if left(r) ∈ Label(left(n))
        then Label(n) ← Label(n) ∪ {r}
  else /* n is a leaf */
    Label(n) ← {all rules that match the operation in n}
```



Productions:

	Production			Cost	Code Template			
1	Goal	→	Assign	0				
2	Assign	→	← (Reg_1,Reg_2)	1	store	r_2	⇒	r_1
3	Assign	→	← (+ (Reg_1,Reg_2),Reg_3)	1	storeAO	r_3	⇒	r_1,r_2
4	Assign	→	← (+ (Reg_1,Num_2),Reg_3)	1	storeAI	r_3	⇒	r_1,n_2
5	Assign	→	← (+ (Num_1,Reg_2),Reg_3)	1	storeAI	r_3	⇒	r_2,n_1
6	Reg	→	Lab_1	1	loadI	l_1	⇒	r_{new}
7	Reg	→	Val_1	0				
8	Reg	→	Num_1	1	loadI	n_1	⇒	r_{new}
9	Reg	→	◆ (Reg_1)	1	load	r_1	⇒	r_{new}
10	Reg	→	◆ (+ (Reg_1,Reg_2))	1	loadAO	r_1,r_2	⇒	r_{new}
11	Reg	→	◆ (+ (Reg_1,Num_2))	1	loadAI	r_1,n_2	⇒	r_{new}
12	Reg	→	◆ (+ (Num_1,Reg_2))	1	loadAI	r_2,n_1	⇒	r_{new}
13	Reg	→	◆ (+ (Reg_1,Lab_2))	1	loadAI	r_1,l_2	⇒	r_{new}
14	Reg	→	◆ (+ (Lab_1,Reg_2))	1	loadAI	r_2,l_1	⇒	r_{new}
15	Reg	→	+ (Reg_1,Reg_2)	1	add	r_1,r_2	⇒	r_{new}
16	Reg	→	+ (Reg_1,Num_2)	1	addI	r_1,n_2	⇒	r_{new}
17	Reg	→	+ (Num_1,Reg_2)	1	addI	r_2,n_1	⇒	r_{new}
18	Reg	→	+ (Reg_1,Lab_2)	1	addI	r_1,l_2	⇒	r_{new}
19	Reg	→	+ (Lab_1,Reg_2)	1	addI	r_2,l_1	⇒	r_{new}
20	Reg	→	- (Reg_1,Reg_2)	1	sub	r_1,r_2	⇒	r_{new}
21	Reg	→	- (Reg_1,Num_2)	1	subI	r_1,n_2	⇒	r_{new}
22	Reg	→	- (Num_1,Reg_2)	1	rsubI	r_2,n_1	⇒	r_{new}
23	Reg	→	× (Reg_1,Reg_2)	1	mult	r_1,r_2	⇒	r_{new}
24	Reg	→	× (Reg_1,Num_2)	1	multI	r_1,n_2	⇒	r_{new}
25	Reg	→	× (Num_1,Reg_2)	1	multI	r_2,n_1	⇒	r_{new}