

قسمت نهم - تولید کد میانی

احمد خادم زاده (khademzadeh@mshdiau.ac.ir)

دانشگاه آزاد مشهد

اصول طراحی کامپیوتر

چرا کد میانی

- چرا مستقیماً کد نهائی تولید نمی کنیم؟
- برخی از ویژگی های مهم کد میانی
 - به سادگی به کد نهائی قابل ترجمه است.
 - به سادگی از کد مبدا قابل تولید است.
 - به راحتی قابل تفسیر و دستکاری است.
- انجام عمل بهینه سازی روی آن راحت تر از کد نهائی انجام می گیرد.
- موجب مستقل از بستر شدن زبان می شود.

مثالی از کد میانی - ماشین پشته ای

- کد میانی جاوا از این گونه است.
- این کد نسبتا کم حجم است.
- مثال

- **A / B + C * D**

Push A

Push B

Div

Push C

Push D

Mul

Add

مثالی از کد میانی – کد سه آدرسه

- فرم نوشتاری آن گونه های متفاوتی دارد
- دستورات سه آدرسه دارای حداکثر سه عملوند و یک عملگر هستند ($A = B \text{ op } C$)
- مثال :

- **$S = B + C / D$**

$$t1 = C / D$$

$$S = B + t1$$

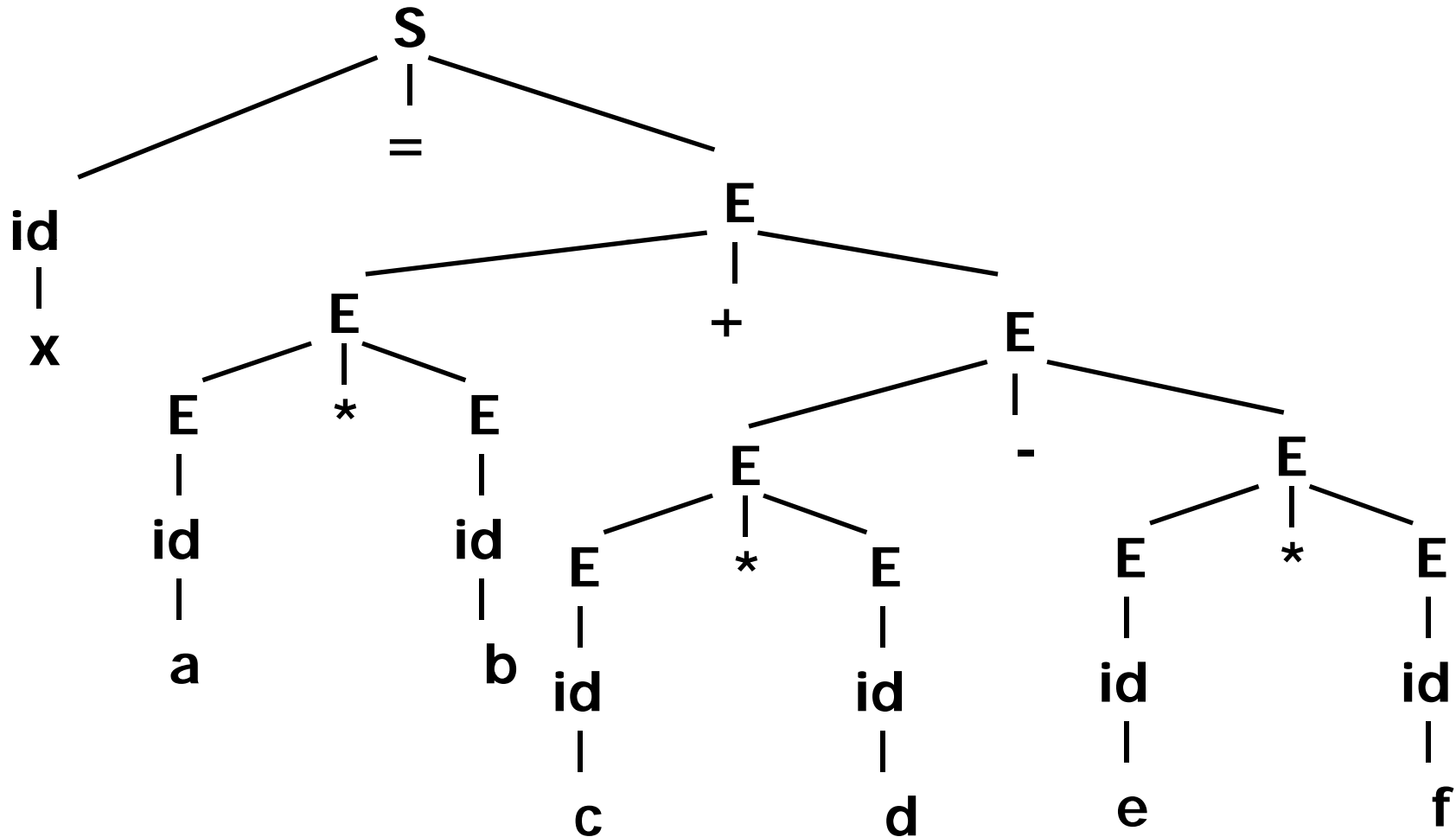
مثال هائی از کد سه آدرسه

- $x = y$
- $x = \text{op } y$
- $x = y \text{ op } z$
- $x = y[i]$
- `goto L`
- `if (x relop y) goto L`
- $y = \text{call } p(x_1, x_2, \dots, x_n)$
- $x = \&y$
- $x = *y$

گرامر و اعمال مربوط به دستور انتساب

```
S → id = E      { p = lookup(id.name);  
                      if (p != NULL) S.code = gen(p '=' E.place);  
                      else error;  
                      S.code = nulllist;  
                      }  
E → E1 + E2    { E.place = newtemp();  
                      E.code = append(E1.code,E2.code,gen(E.place '=' E1.place '+' E2.place));  
                      }  
E → E1 * E2    { E.place = newtemp();  
                      E.code = append(E1.code,E2.code,gen(E.place '=' E1.place '*' E2.place));  
                      }  
E → - E1        { E.place = newtemp();  
                      E.code = append(E1.code,gen(E.place '=' '-' E1.place));  
                      }  
E → (E1)        { E.place = E1.place; E.code = E1.code; }  
E → id          { p = lookup(id.name);  
                      if (p != NULL) E.place = p;  
                      else error;  
                      E.code = nulllist;  
                      }
```

$$x = a * b + c * d - e * f$$

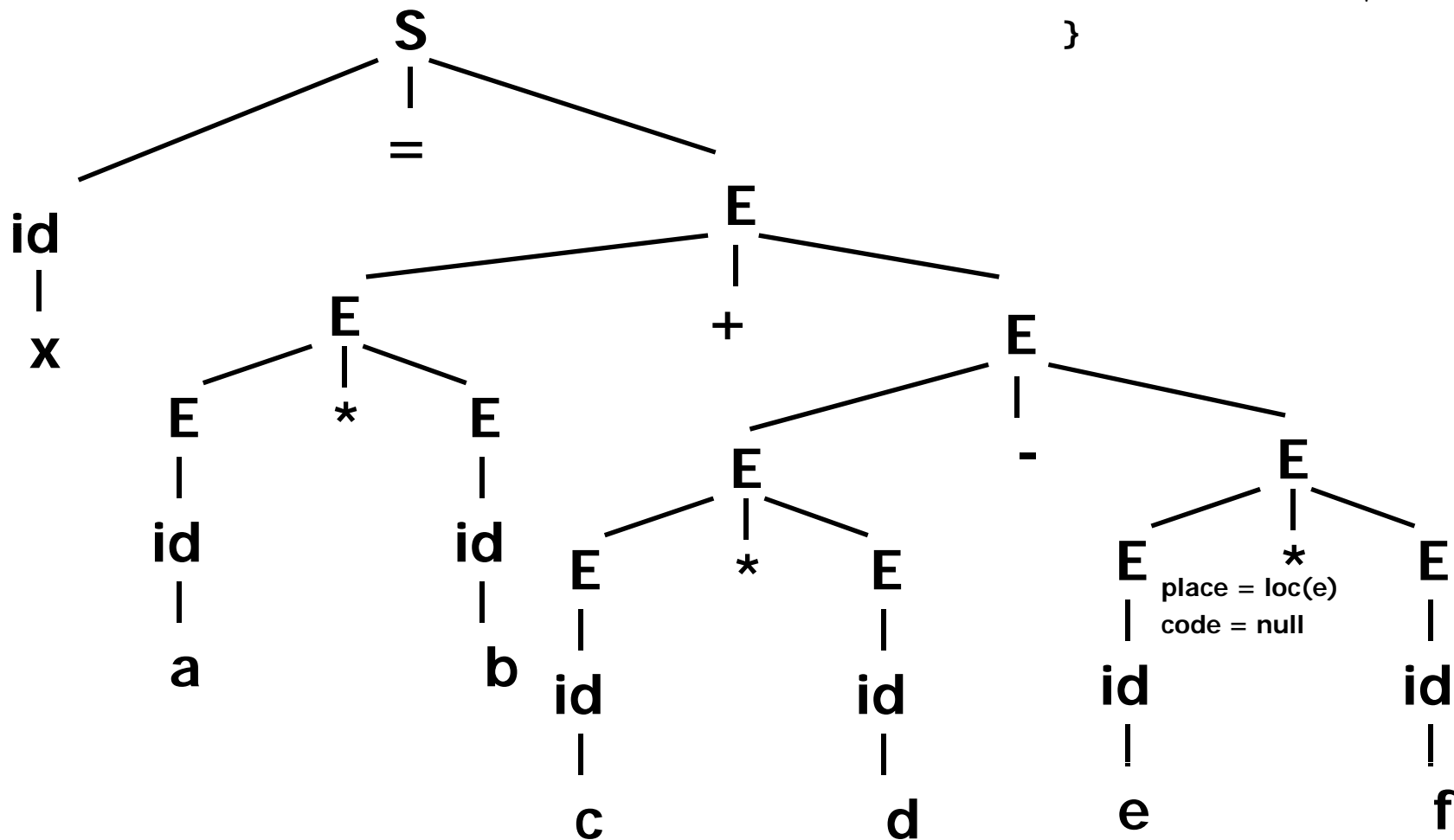


$$x = a * b + c * d - e * f$$

```

E → id
{ p = lookup(id.name);
  if (p != NULL) E.place = p;
  else error;
  E.code = nulllist;
}

```

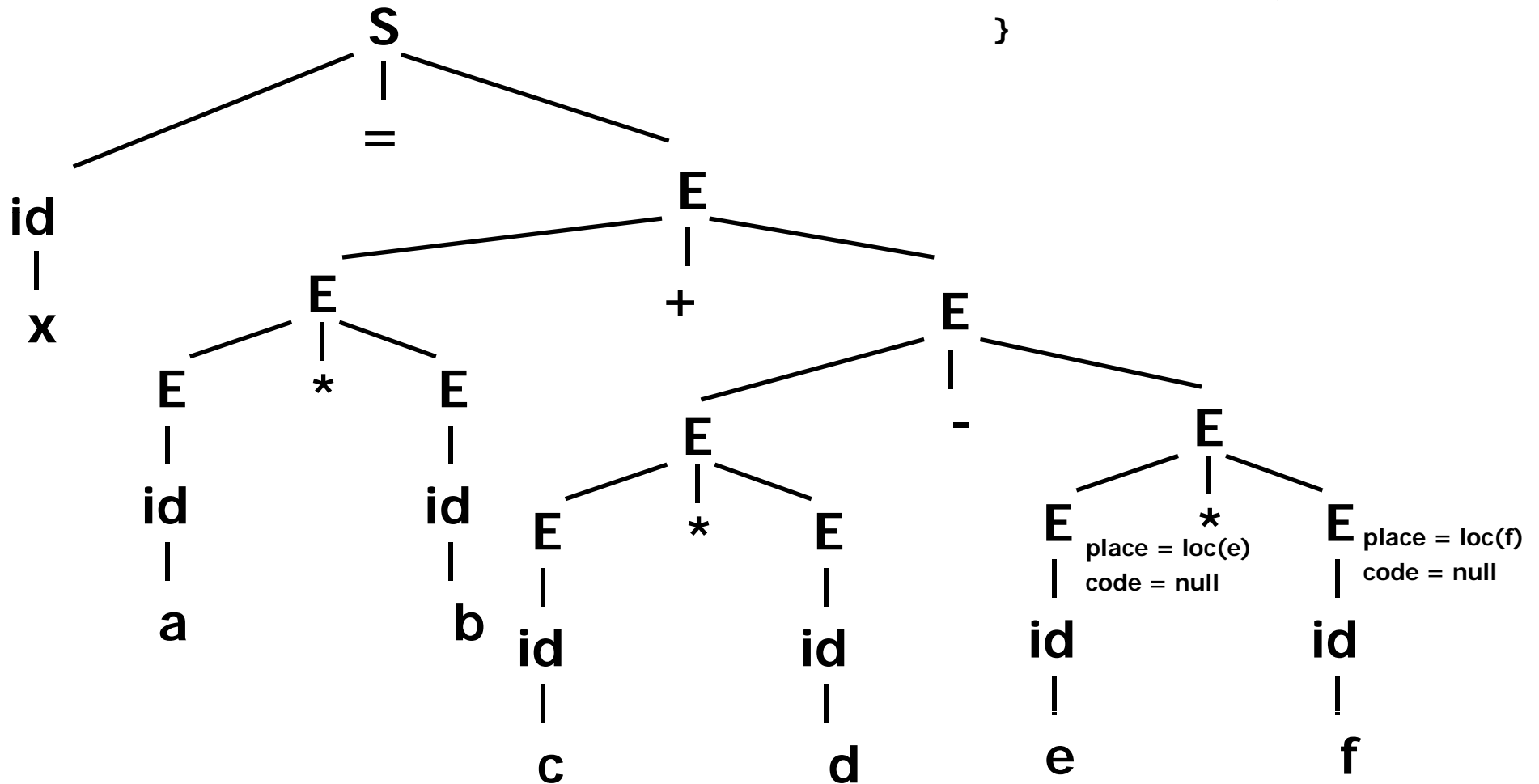


$$x = a * b + c * d - e * f$$

```

E → id
{ p = lookup(id.name);
  if (p != NULL) E.place = p;
  else error;
  E.code = nulllist;
}

```



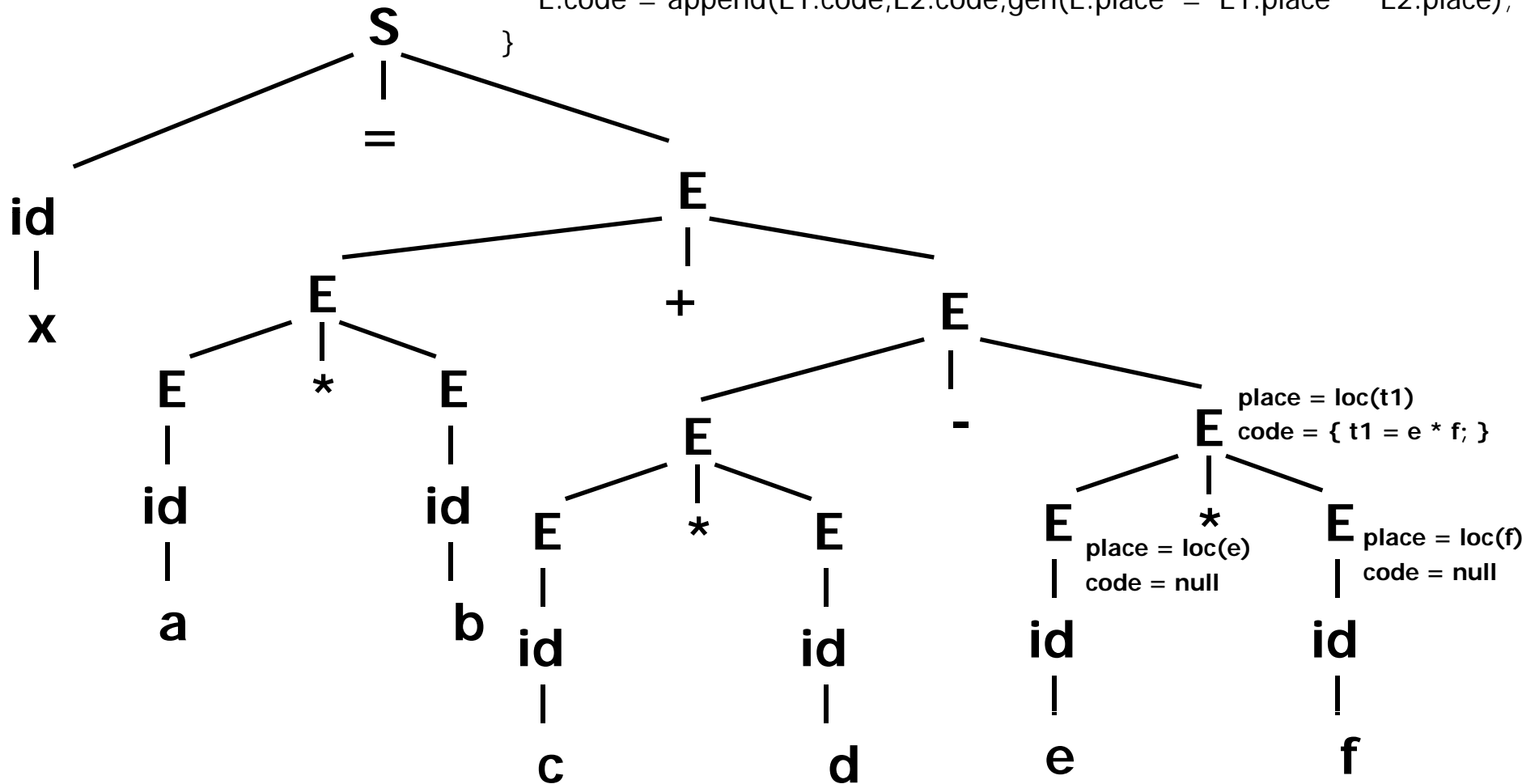
$$x = a * b + c * d - e * f$$

$E \rightarrow E * E$

```
{ E.place = newtemp();
```

```
  E.code = append(E1.code, E2.code, gen(E.place '=' E1.place '*' E2.place));
```

```
}
```



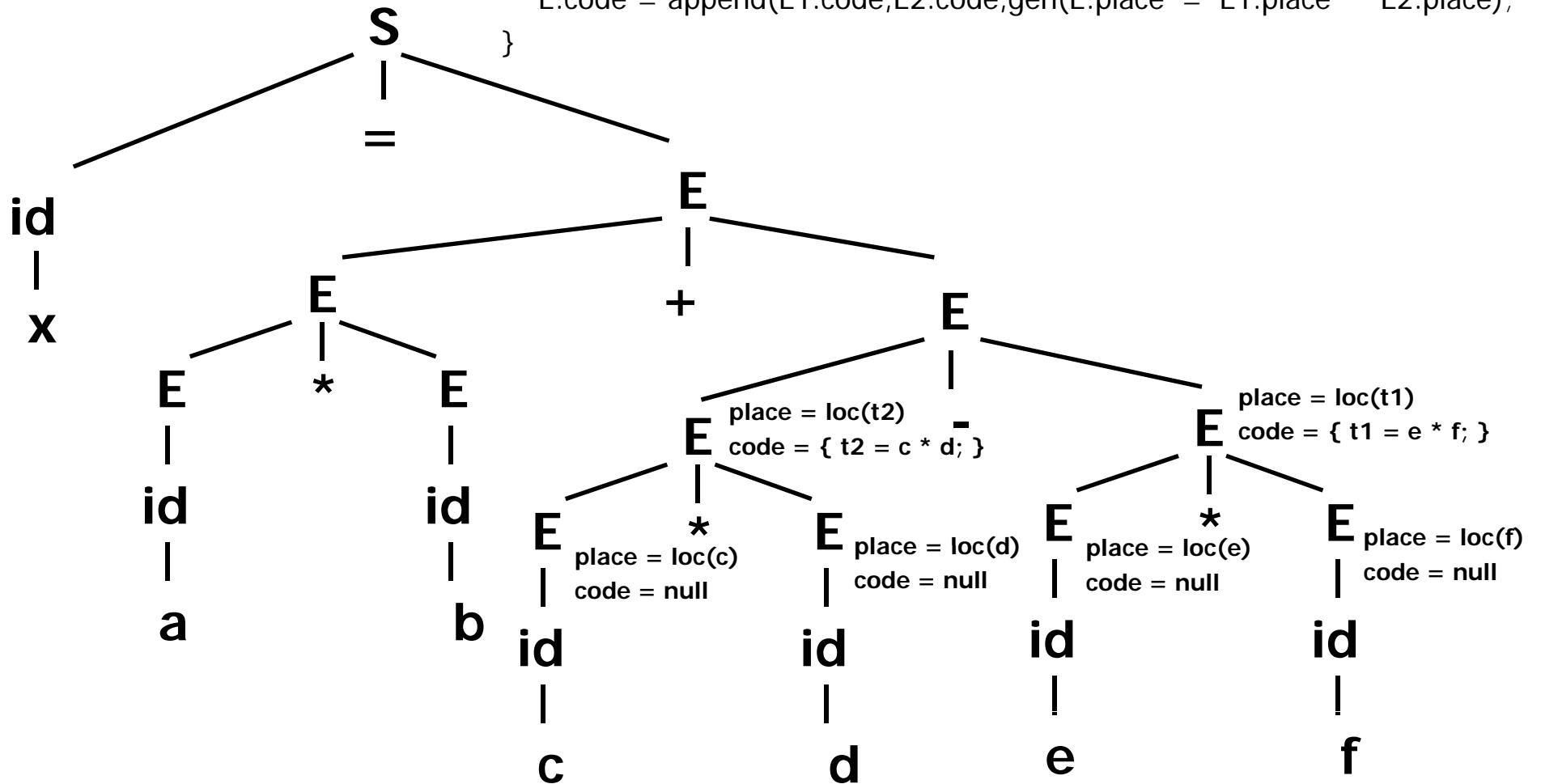
$$x = a * b + c * d - e * f$$

$E \rightarrow E * E$

```
{ E.place = newtemp();
```

```
  E.code = append(E1.code, E2.code, gen(E.place '=' E1.place '*' E2.place));
```

```
}
```



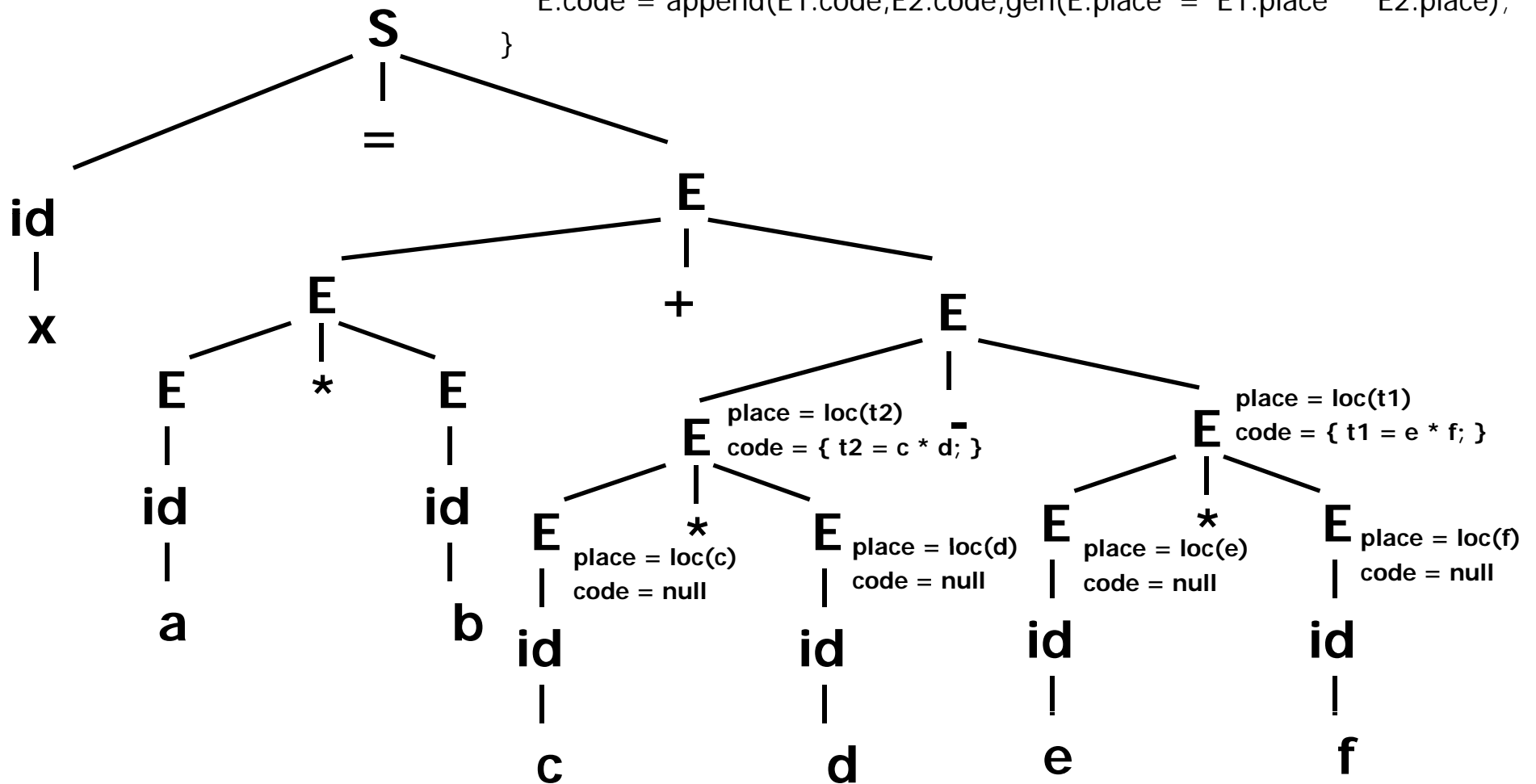
$$x = a * b + c * d - e * f$$

$E \rightarrow E * E$

{ E.place = newtemp();

E.code = append(E1.code, E2.code, gen(E.place '=' E1.place '*' E2.place);

}



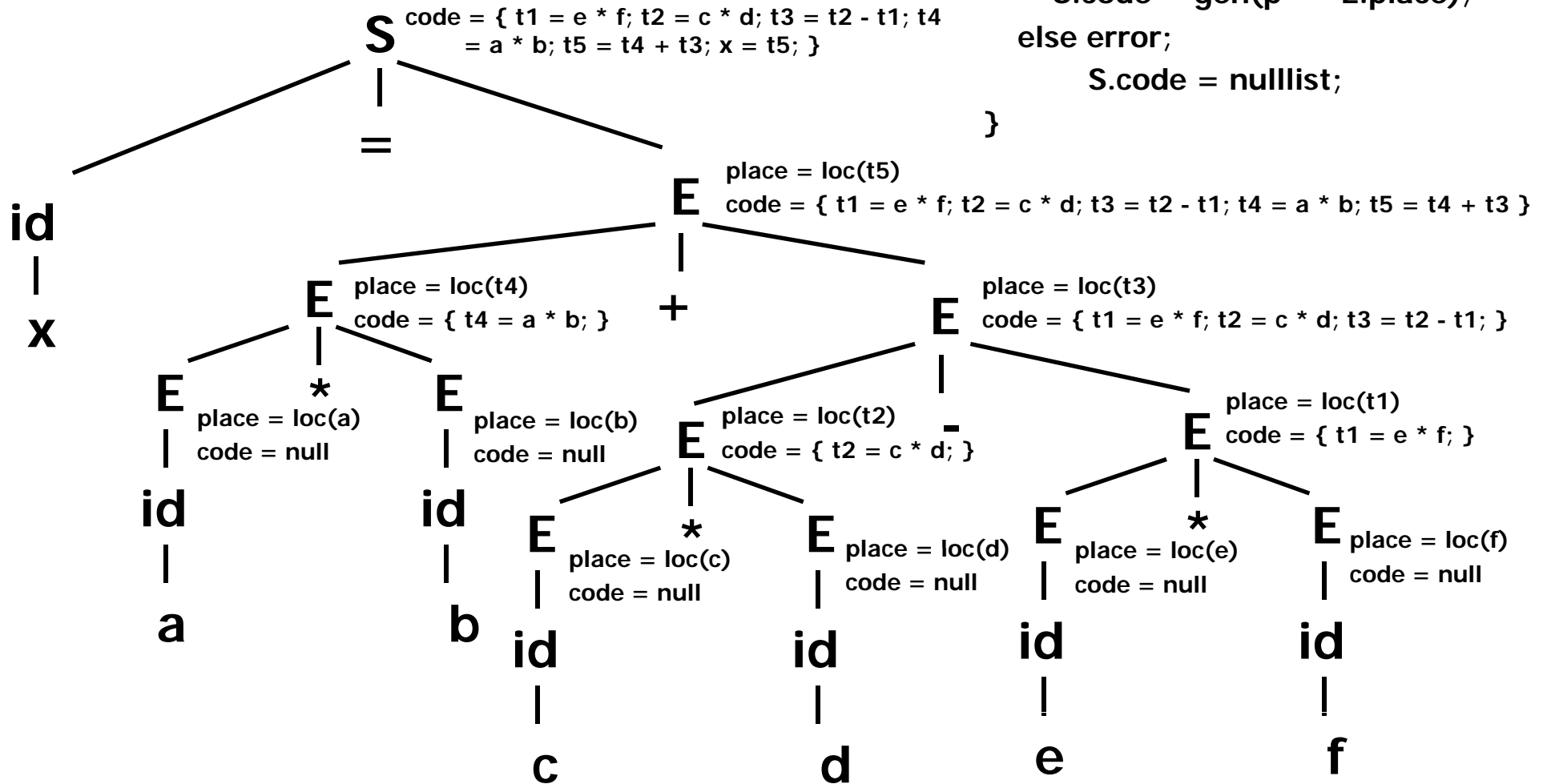
$$x = a * b + c * d - e * f$$

$S \rightarrow id = E$

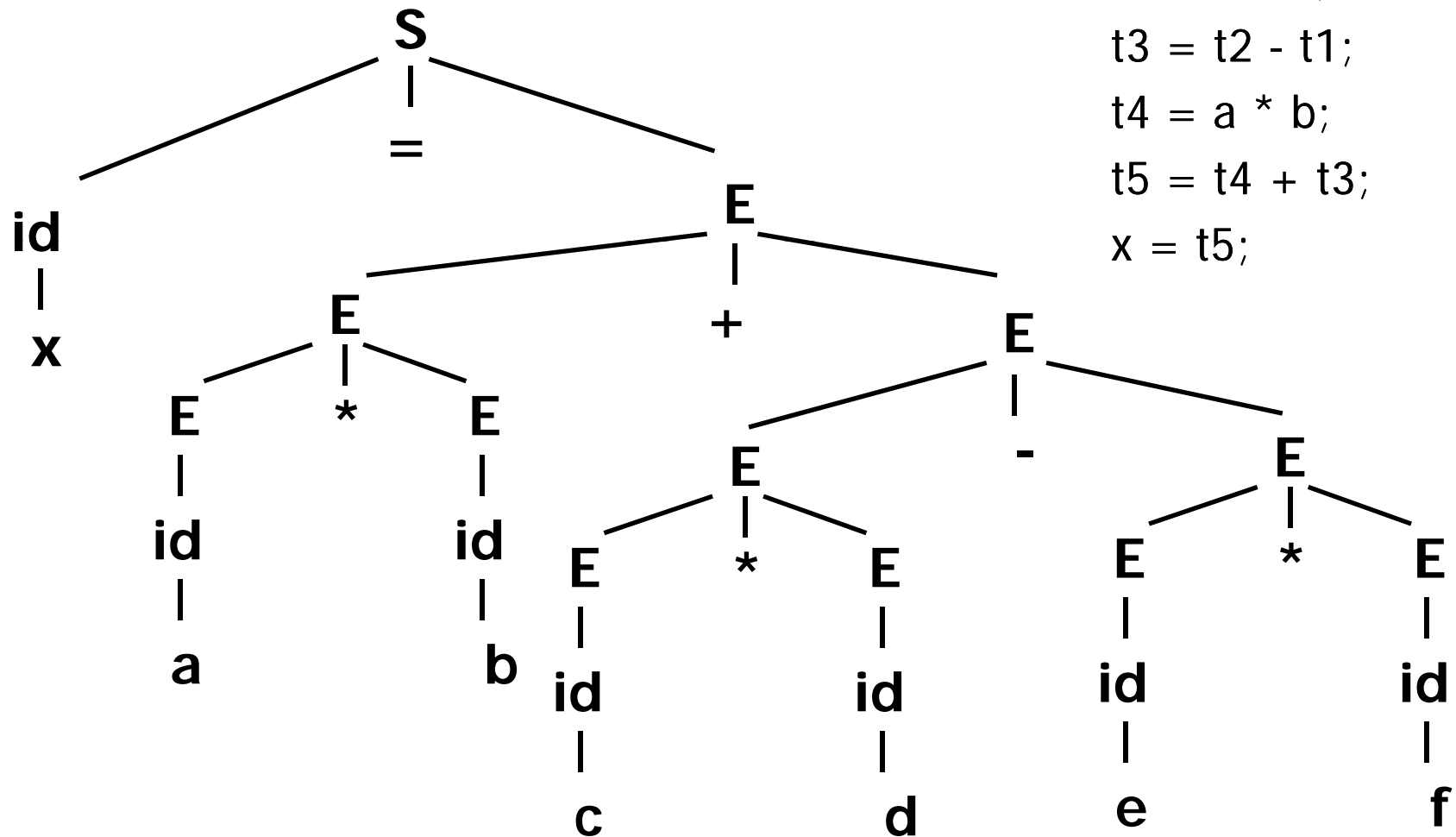
```

{
  p = lookup(id.name);
  if (p != NULL)
    S.code = gen(p != E.place);
  else error;
  S.code = nulllist;
}

```



$$x = a * b + c * d - e * f$$



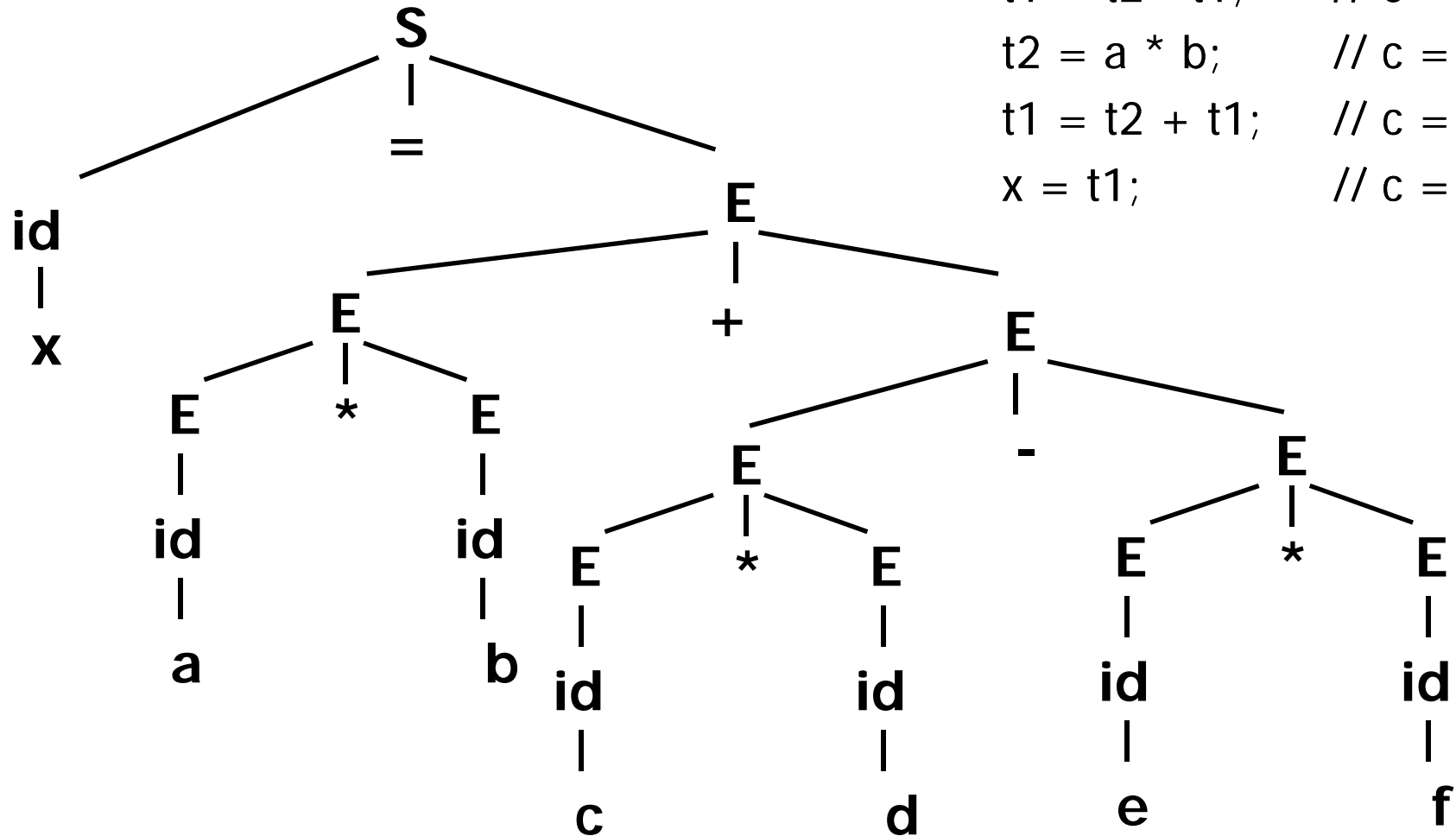
t1 = e * f;
 t2 = c * d;
 t3 = t2 - t1;
 t4 = a * b;
 t5 = t4 + t3;
 x = t5;

$$x = a * b + c * d - e * f$$

```

// c = 0
t1 = e * f; // c = 1
t2 = c * d; // c = 2
t1 = t2 - t1; // c = 1
t2 = a * b; // c = 2
t1 = t2 + t1; // c = 1
x = t1; // c = 0

```



دسترسی به عناصر آرایه

- آرایه سطری دو بعدی $A[\text{low}_1..\text{high}_1, \text{low}_2..\text{high}_2]$ با آدرس شروع S و اندازهٔ عناصر W
- آدرس عنصر $A[i,j]$:

- $S + (i - \text{low}_1)(\text{high}_2 - \text{low}_2 + 1) \times w + (j - \text{low}_2) \times w$
 - $\text{Length}_2 = (\text{high}_2 - \text{low}_2 + 1)$
- $S + i \times \text{Length}_2 \times w + j \times w - (\text{low}_1 \times \text{Length}_2 \times w) + (\text{low}_2 \times w)$
 - $S_0 = S - (\text{low}_1 \times \text{Length}_2 \times w) + \text{low}_2 \times w$
- $S_0 + (i \times \text{Length}_2 + j) \times w$

دسترسی به عناصر آرایه

```
L → Elist ]      { L.place = newtemp();  
                   L.offset = newtemp();  
                   code1 = gen(L.place '=' constTerm(Elist.array));  
                   code2 = gen(L.offset '=' Elist.place  
                               *sizeof(baseType(Elist.array)));  
                   L.code = append(Elist.code,code1,code2);  
                   }
```

```
Elist → Elist1 , E { t = newtemp();  
                       m = Elist1.ndim + 1;  
                       code1 = gen(t = Elist1.place * numElem(Elist1.array,m));  
                       code2 = gen(t = t + E.place);  
                       Elist.array = Elist1.array;  
                       Elist.place = t;  
                       Elist.ndim = m;  
                       Elist.code = append(Elist1.code,E.code,code1,code2);  
                       }
```

```
Elist → id [ E    { Elist.array = id.place;  
                   Elist.place = E.place;  
                   Elist.ndim = 1;  
                   Elist.code = E.code;  
                   }
```

دسترسی به عناصر آرایه

```
E → L      { if (L.offset = NULL) then E.place = L.place;
                else E.place = newtemp;
                E.code = gen(E.place = L.place[L.offset]);
            }
S → L = E   { if L.offset = NULL then E.code = gen(L.place = E.place);
                else S.code = append(E.code,gen(L.place[L.offset] = E.place);
            }
L → id     { L.place = id.place;
                L.offset = null;
            }
```

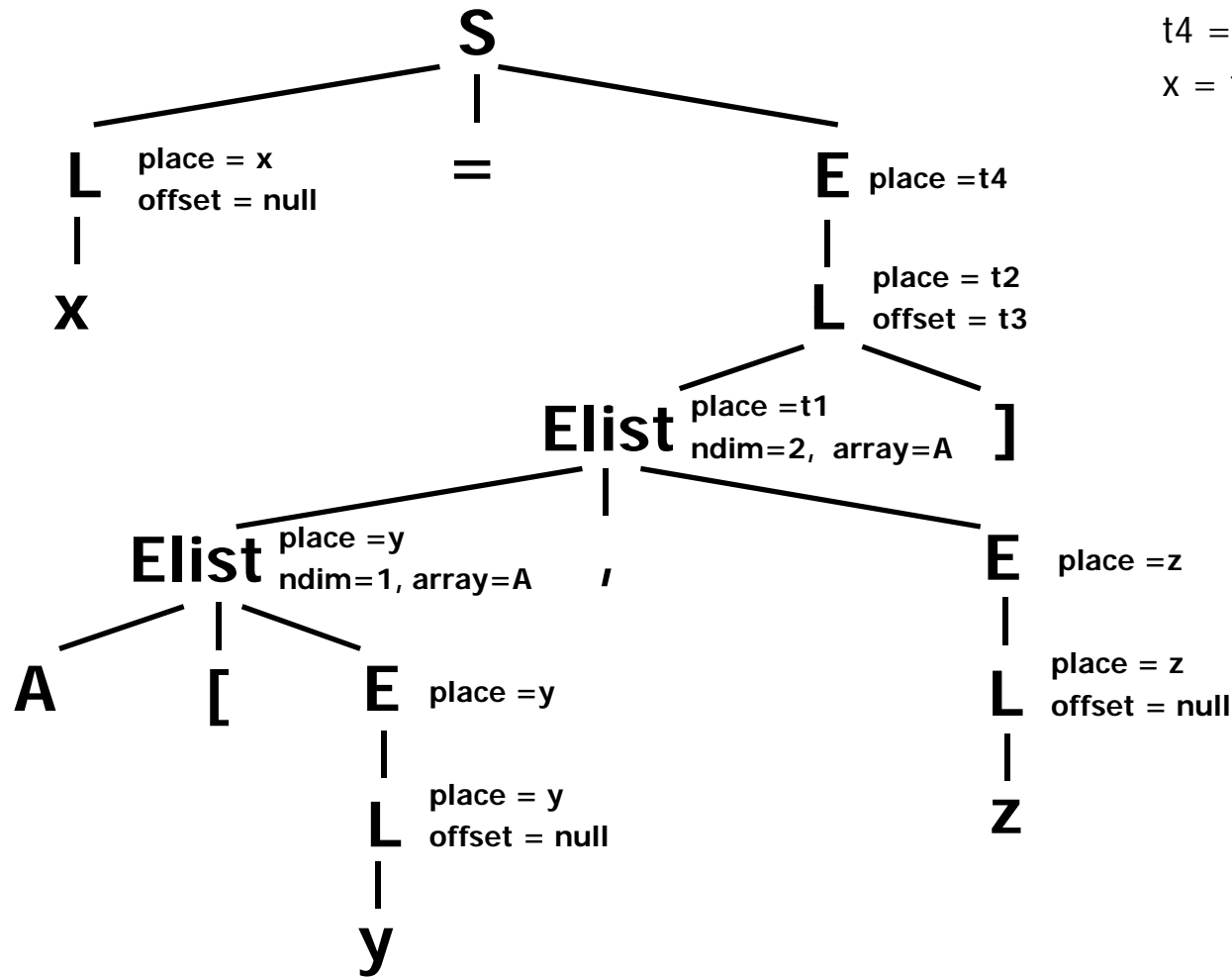
- مثال : آرایه $A[1..10, 1..20]$ که اندازه عناصر آن چهار بایت است؛ تولید کد برای $x=A[y,z]$ را انجام می دهیم.

$x = A[y, z]$

```

t1 = y * 20
t1 = t1 + z
t2 = c
t3 = t1 * 4
t4 = t2[t3]
x = t4

```



عبارات منطقی و رابطه ای

```
E → false      { E.place = newtemp()
                   E.code = {gen(E.place = 0)}
                   E.laststat = E.nextstat + 1
                   }
E → true       { E.place = newtemp()
                   E.code = {gen(E.place = 1)}
                   E.laststat = E.nextstat + 1
                   }
E → (E1)      { E.place = E1.place;
                   E.code = E1.code;
                   E1.nextstat = E.nextstat
                   E.laststat = E1.laststat
                   }
E → not E1   { E.place = newtemp()
                   E.code = append(E1.code,gen(E.place = not E1.place))
                   E1.nextstat = E.nextstat
                   E.laststat = E1.laststat + 1
                   }
```

- `if(x!=0 && y/x>2) then ...`
- `orElse` `or`
- `andThen` `and`

عبارات منطقی و رابطه ای

E → E1 or E2	<pre>{ E.place = newtemp() E.code = append(E1.code,E2.code,gen(E.place = E1.place or E2.place) E1.nextstat = E.nextstat E2.nextstat = E1.laststat E.laststat = E2.laststat + 1 }</pre>
E → E1 and E2	<pre>{ E.place = newtemp() E.code = append(E1.code,E2.code,gen(E.place = E1.place and E2.place) E1.nextstat = E.nextstat E2.nextstat = E1.laststat E.laststat = E2.laststat + 1 }</pre>
E → id1 relop id2	<pre>{ E.place = newtemp() E.code = gen(if id1.place relop id2.place goto E.nextstat+3) E.code = append(E.code,gen(E.place = 0)) E.code = append(E.code,gen(goto E.nextstat+2)) E.code = append(E.code,gen(E.place = 1)) E.laststat = E.nextstat + 4 }</pre>

مثال

• برای عبارت منطقی زیر کد تولید کنید:

$M < N$ and $P < Q$ or $R < S$

دستورات شرطی و حلقه while

S → if E then S₁	{ E.true = newlabel E.false = S.next S ₁ .next = S.next S.code = append(E.code,gen(E.true:),S ₁ .code) }
S → if E then S₁ else S₂	{ E.true = newlabel E.false = newlabel S ₁ .next = S.next S ₂ .next = S.next S.code = append(E.code,gen(E.true:),S ₁ .code, gen(goto S.next),gen(E.false :),S ₂ .code) }
S → while E do S₁	{ S.begin = newlabel E.true = newlabel E.false = S.next S ₁ .next = S.begin S.code = append(gen(S.begin:),E.code,gen(E.true:),S ₁ .code, gen(goto S.begin) }