Generación de código intermedio

Instrucciones aritméticas

$$x = z+2*8-(a/b)-x$$

t0 = 2 * 8
t1 = a / b
t2 = t0 – t1
t3 = z –x
t4 = t2 + t3
x = t4
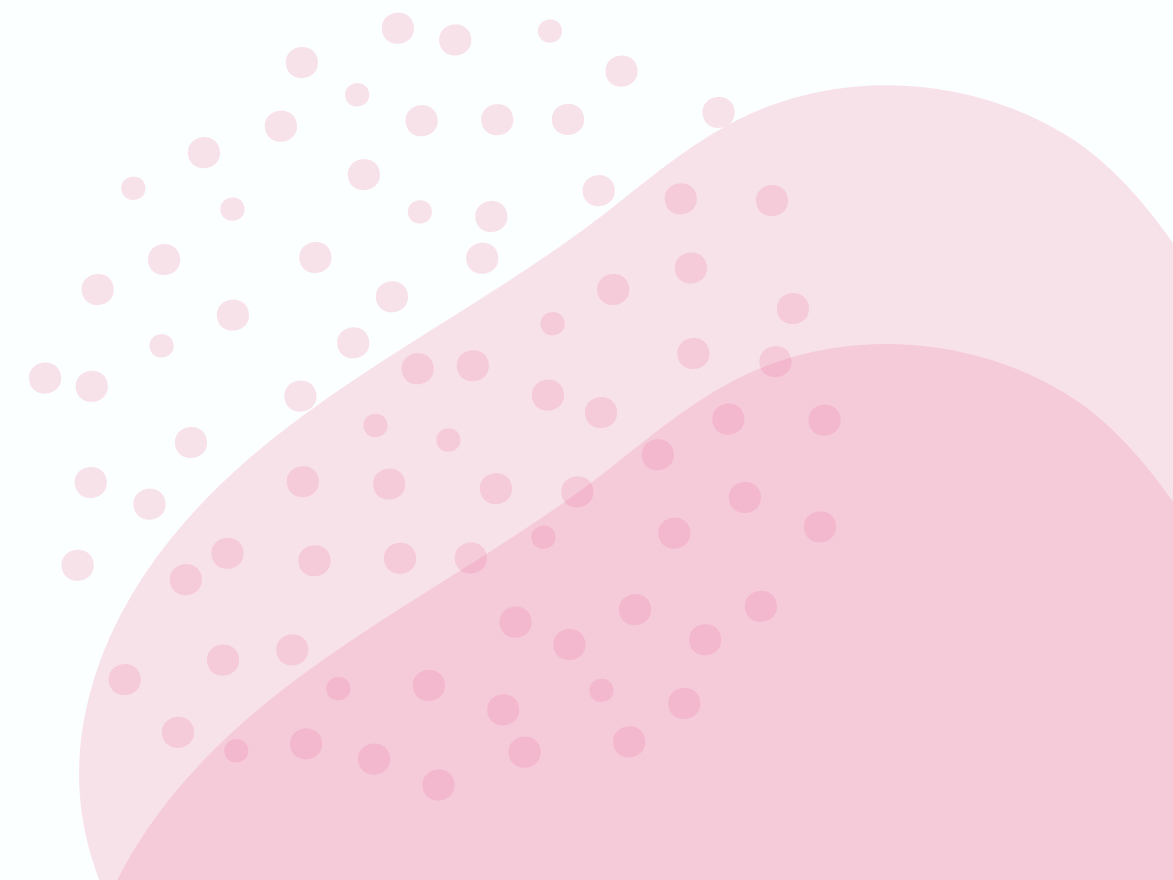
```
int x[6];
x[j+j*2]= a*3/6
```

t0 = 3/6
t1 = a*t0
t2 = j*2
t3 = t2+j
t4 = t3*4
x[t4] = t1

```
int x[6], y[5][3][2];
x[j+j*2] = y[i+2][j-1][k*3]
```

t0 = i+2

t1 = t0*24

t2 = j-1

t3 = t2*6

t4 = t1+t3

t5 = k*3

t6 = t5*4

t7 = t4+t6

t8 = y[t7]

t9 = j*2

t10 = t9+j

t11 = t10*4

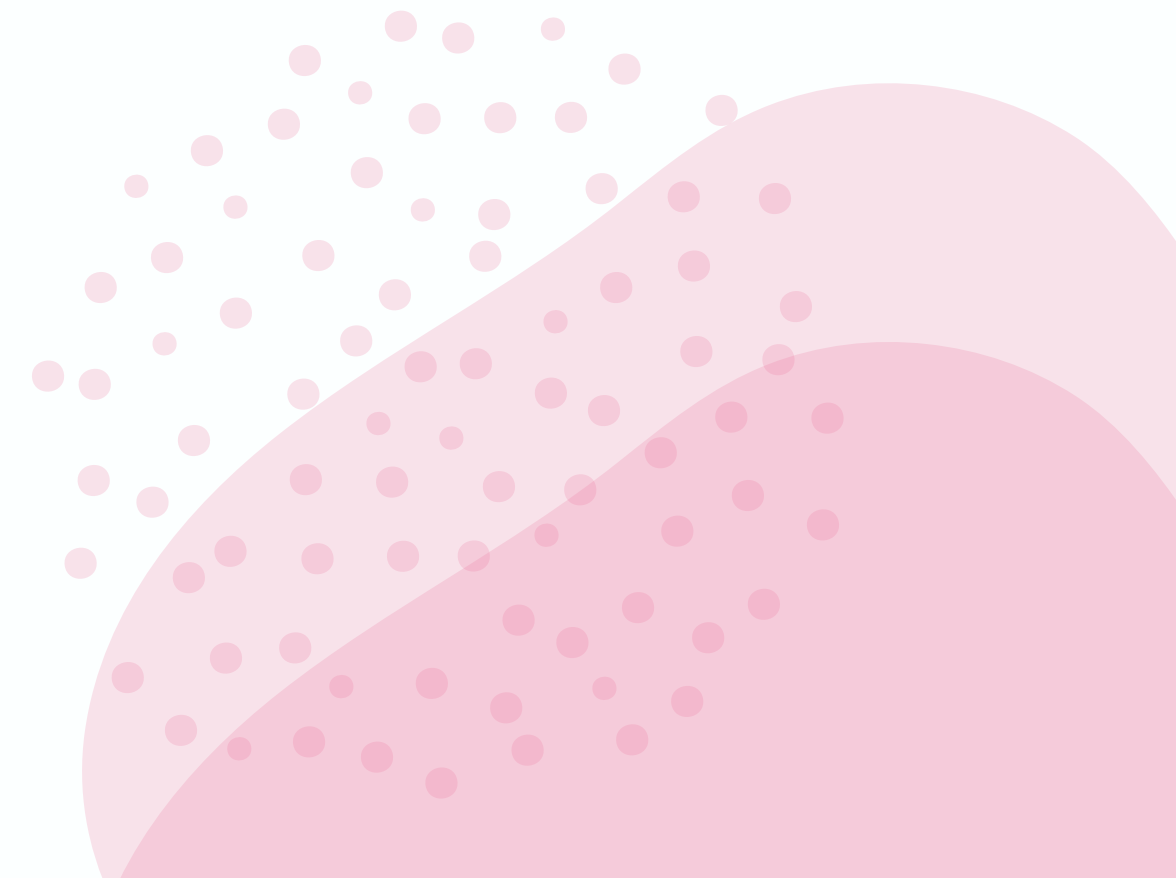x[t11] = t8

Generación de código intermedio

Instrucciones if,if-else,while y do-while

```
if(x[i+2]>x[i-1])
    a = z*3+y;
else
    a = 2;
```

```
t0 = i-1
t1 = t0*4
t2 = x[t1]
t3 = i+2
t4 = t3*4
t5 = x[t4]
if t5 > t2  goto L0
        goto L1


L0: t6 = z*3
    t7 = t6+y
    a = t7


L1: a = 2
```

```
while(x[i+2]>x[i-1])
{
    if(y<3)
        y=1;
    else
        y=0;
}
```

```
L0:    t0 = i-1
       t1 = t0*4
       t2 = x[t1]
       t3 = i+2
       t4 = t3*4
       t5 = x[t4]
       if t5>t2 goto L1
          goto L2

L1:    if y<3 goto L3
          goto L4

L2: ...

L3:    y=1
       goto L0

L4:  y=0
     goto L0
```

```
do
    i++;
    j++;
while(x>z*5)
```

L0:   t0 = i+1
        i = t0
        t1 = z*5
        if x > t1 goto L0
            goto L1

L1: ...