```
// A:=A+B (soma não-destrutiva)
operation soma(A,B,C){
  1: if zero B then goto 5 else goto 2
  2: do dec B goto 3
  3: do inc A goto 4
  4: do inc C goto 1
  5: if zero C then goto 0 else goto 6
  6: do dec C goto 7
  7: do inc B goto 5
}
```

```
// limpa variável
operation clear(A){
 1: if zero A then goto 0 else goto 2
 2: do dec A goto 1
}
```

```
// A:=B usando C
operation load(A,B,C){
  1: do clear(A) goto 2
  2: do soma(A,B,C) goto 0
}
```

```
// A:=fst(B) usando C,D
operation fst(A,B,C,D){
  1: do clear(A) goto 2
  2: do load(C,B,D) goto 3
  3: if divBy2(C,D) then
goto 4 else goto 6
  4: do div2(C,D) goto 5
  5: do inc A goto 3
  6: do clear(C) goto 0
}

// A:=snd(B) usando C,D
operation snd(A,B,C,D){
  1: do clear(A) goto 2
  2: do load(C,B,D) goto 3
  3: if divBy3(C,D) then
goto 4 else goto 6
  4: do div3(C,D) goto 5
  5: do inc A goto 3
  6: do clear(C) goto 0
}

main {
  1: do fst(A,X,C,D) goto 2
  2: do snd(B,X,C,D) goto 3
  3: do soma(Y,A) goto 4
  4: do soma(Y,B) goto 0

}
```

```
// A:=A div 3 usando C
operation div3(A,C){
  1: do load(C,A) goto 2
  2: if zero C then goto 0 else goto 3
  3: do dec C goto 4
  4: if zero C then goto 0 else goto 5
  5: do dec C goto 6
  6: if zero C then goto 0 else goto 7
  7: do dec C goto 8
  8: do inc A goto 2
}
// A divisivel_por_3 usando C
test divBy3(A,C){
  1: do load(C,A) goto 2
  2: if zero C then goto true else
goto 3
  3: do dec C goto 4
  4: do inc A goto 5
  5: if zero C then goto false else
goto 6
  6: do inc A goto 7
  7: do dec C goto 8
  8: if zero C then goto false else
goto 9
  9: do dec C goto 10
 10: do inc A goto 2
}
```

```
// A divisivel_por_2 usando C
test divBy2(A,C){
  1: do load(C,A) goto 2
  2: if zero C then goto true
else goto 3
  3: do dec C goto 4
  4: do inc A goto 5
  5: if zero C then goto false
else goto 6
  6: do inc A goto 7
  7: do dec C goto 2
}
// A:=fst(B) usando C,D
operation fst(A,B,C,D){
  1: do clear(A) goto 2
  2: do load(C,B,D) goto 3
  3: if divBy2(C,D) then goto
4 else goto 6
  4: do div2(C,D) goto 5
  5: do inc A goto 3
  6: do clear(C) goto 0
}
```

```
// B := A * C usando D (atribuição não
destrutiva)
operation multiplicacao(A,B,C,D){
  1: if zero C then goto 3 else goto 2
  2: if zero A then goto 3 else goto 4
  3: do clear(B) goto 0
  4: if zero C then goto 0 else goto 5
  5: do soma(B,A,D) goto 6
  6: do dec C goto 4
}

// Y:=A+B usando C (soma
conservativa)
operation somaNova(Y,A,B,C){
   1: do atribui(Y,A,C) goto 2
   2: if zero B then goto 6 else goto 3
   3: do dec B goto 4
   4: do inc Y goto 5
   5: do inc C goto 2
   6: if zero C then goto 0 else goto 7
   7: do dec C goto 8
   8: do inc B goto 6
}

// fibonacci
main {
   1: do atribui(C,X,A) goto 2
   2: if zero C then goto 0 else goto 3
   3: do dec C goto 4
   4: do inc M goto 5
   5: do atribui(Y,M,D) goto 6
   6: if zero C then goto 0 else goto 7
   7: do somaNova(Y,M,N,D) goto 8
   8: do atribui(N,M,D) goto 9
   9: do atribui(M,Y,D) goto 10
   10: do dec C goto 6
}
```

```
 true = \x y. x;
 false = \x y. y;
 if = \b e1 e2. b e1 e2;
 not = \b. b false true;
 and = \b1 b2. if b1 b2 false;
 or = \b1 b2. if b1 true b2;
 succ = \n. \A B. A (n A B);
 add = \n1 n2. n1 succ n2;
 mult = \n1 n2. n1 (\x. add n2 x) 0;
 expo = \n1 n2. N2 (\x. mult n1 x) 1
 isZero = \n. n (\x. false) true;
 pair = \a b. \c. c a b;
 fst = \p. p true;
 snd = \p. p false;
 swap = \p. pair (snd p) (fst p);
 shiftInc = \p. pair (snd p) (succ (snd p));
 pred = \n. fst (n shiftInc (pair 0 0));
 sub = \a b. b pred a;
 -- (menor a b) testa a < b
 menor = \a b. (\n. not (isZero n)) (sub b a);
 maior = \a b. Menor b a
 diferente = \a b. Or (menor a b) (maior a b)
 -- Listas
 empty = \x. true;
 cons = pair;
 isEmpty = \l. l (\a b. false);
 head = fst;
 tail = snd;
 --Fat
 Y = \f. (\x. f(x x)) (\x. f(x x))
 padrao_fat = \R n. (isZero n) 1 (mult n (R (pred n)))
 Slen = \R l. if (isEmpty l) 0 (succ (R (tail l)))
 Ssum = \R l. if (isEmpty l) 0 (add (head l) (R (tail l)))
 Fat = Y padrao_fat
 Len = Y slen
 Sum = Y Ssum
```