```
1: faça F vá_para 2
2: se T1 então vá_para 1 senão vá_para 3
3: faça G vá_para 4
4: se T2 então vá_para 5 senão vá_para 1
```

P é R

R def F; (Se T1 R senão S)

S def G; se T2 encerra sendo R)

Fatorial: f= f *n-1
3!= 3*2*1

1: faça ler entrada N vá para 2:
2: se N == 0 vara para 9 senao va para 3
3: se N==1 vara para 9 senao va para 4
4: faça F:=N vá para 5
5: faça N:= N-1 vá para 6
6: faça F:= F * N vá para 7
7: se N = 1 vá para 8 senao va para 5
8: print F va para 200
9: print 1 va para 200

Cálculo Lambda e Funções Recursivas de Kleene

1) λxx ou λx.x
λxM é um termo lambda
2) λx NÃO é um termo lambda
3)

$\lambda x \lambda yx$
x variavel: termo lambda: M
$\lambda x \lambda y$M : λyM : M
λxM: termo lambda

λxM e (F A) = λxMA simplificação/redução

$$(\lambda xx(yz)) \Rightarrow (yz)$$

(λxx(yz)) = (λx M=x A=(yz)

1(λxxλxx) ⇒ λxx;

2. ((λxλy(xy)λxx)x) ⇒ (λy(λxxy)x) ⇒ (λxxx) ⇒ x;

3. (λx(xx)λx(xx)) ⇒ (λx(xx)λx(xx)) (irredut´ıvel);

4. (λxyz) ⇒ y (jogar fora alguma coisa).

1(λxxλxx) ⇒ λxx;

1(λxMλxM) ⇒ λxx;

1(FA) ⇒ λxx;

2. ((λxλy(xy)λxx)x) =
𝜆xM e (F A) = 𝜆xMA
λy(xy) λxx = xλxx
F A
((λxxλxx) x) = (λxxx) =
F A

(λxxx) = x (resposta)
F A

Outra forma de resolver:
2. ((λxλy(xy)λxx)x) =
F A
(λy(λxxy)x) = λxxx
F A

λxxx = **𝜆xMA**
F. A
= x (resposta)

𝜆xMA
3. (λx(xx)λx(xx)) ⇒ (λx(xx)λx(xx)) (irredutível);

(λx(xx)λx(xx)) = (λx(xx) λx(xx)) resposta
F. A

4. (λxyz) ⇒ y (jogar fora alguma coisa).

(λxyz) = y resposta
F. A

𝜆xMA = FA
1. (λz(λyzx)(xx)) = λzz(xx) = (xx) ou xx
F. A

(λy(xx)x) = xx ou (xx)
F. A

2. (λxxλxx) = λxx

3. (λx(xx)λyy(xx))=
F. A
(λyy(xx)λyy(xx)) =
F. A
y(xx) = M
(λyy(xx)(xx))=
F. A
y(xx) = M
(xx)(xx) = resposta

———————————-

λxMA = FA
λxyz=

λxyz= y Sobra só o corpo M

——————————————-

Avaliação de expressão Lambda:
λxy.(+xy) 3 4 OU λxλy.(+xy) 3 4
λxy. É a mesma coisa: λxλy
Avaliar x e seu argumento 3:
λy.(+ 3 y) 4
Avaliar y e seu argumento 4:
(+ 3 4) equivale 3 + 4
Avaliar a expressão: soma de 3 + 4 = 7
7

————————————-

If A then B else C
((T a)b) ≡ ((λxλyxa)b)
((λxλyxa) b)
λyxa = x
F. A

((λxx) b) = b
———- outra forma de avaliar:
((λxλyxa)b) =
((λxλyxa)b)=
F. A
λyba= b
F. A
———————

((Fa)b) ≡ ((λxλyya)b)
((λxλyya)b)
((λxλyya)b) = λyya = a

———————

not ≡ (λx((xF)T)
(notF) = T = λxλyx

(λx((xF)T)F)=

(λx((xF)T)F)= ((FF)T)=
((FF)T)= ((λxλyy λxλyy) T) =
(λyy T) = T=λxλyx

—>
Heap: 80, 30, 60, 50, 70, 20, 40
Objetos: 32, 64, 48, 16

First Fit: 32 -> 80, 64 -> 70, 48 -> 60, 16 -> 30
Best Fit: 32->40, 64 ->70, 48 ->50, 16 ->20
Worst Fit: 32->80 , 64 ->70 , 48 ->60 , 16 ->50

Kleene:
Id(x) = x
id(0)=0
id(1)=1
id(5)=5
zero(x) = 0
zero(x) = id(0)=0
s(x)=x+1
p(x) = x-1, x>0, senão 0

s(0) = 0 +1 = 1
s(1)= 1+1 = 2
s(2)= 2+1=3
p(0) = 0
p(1) = 1-1=0
p(2)= 2 -1 = 1
p32(10,20,30) = 20
p33(s(1) , p(s(3)), s(p(4))) = s(p(4)) = s(3)=4
s(s(s(0))) = s(s(1))= s(2)= 3
p32(s(0), p(0), p22(s(3), p(0))) = p(0)=0
p33(s(0), p(0), p22(s(3), p(0)))= p22(s(0), p(0))=p(0)=0

Função constante um , fum(x)=1
fum(x)=id(1)=1
fum(x)=s(zero(x))=s(0)=1
fdois(x)=s(fum(x))=s(1)=2
fdois(x)=s(s(zero(x)))=s(s(0))=s(1)=2

soma2(x)= x + 2

soma2(x)= s(s(x))
soma3(x)= s(s(s(x))) = x + 3
soma3(x)= s(soma2(x))
soma3(x) = p22(s(0), s(s(s(x)))) = s(s(s(x)))

Condicional: cond(b, g1, g2) = se b entao g1 senao g2
Ex: se a > 2 , s(a) , zero(a)
logica(a) = cond(maior(a,2) , s(a) , zero(a))

maior(x,y) = cond(x > y, x , y)

zero(x) = cond(x=0, p11(x), p22(p(x), zero(p(x))))
zero(2) = cond(x=0, p11(2), p22(p(2) , zero(p(2))))
p22(p(2) , zero(p(2))) =
zero(p(2)) = zero(1) = cond(x=0, p11(1) , p22(p(1), zero(p(1))))
p22(p(1), zero(p(1))) =
zero(p(1)) = zero(0) = cond(x=0, p11(0) , p22(p(0), zero(p(0))))
zero(0) = p11(0) = 0

Recursão Primitiva:

$$f(x_1,\ldots,x_n,y) = \begin{cases} g(x_1,\ldots,x_n) & \text{se } y = 0 \\ h(x_1,\ldots,x_n,p(y),f(x_1,\ldots,x_n,p(y))) & \text{se } y \neq 0 \end{cases}$$

https://www.cin.ufpe.br/~ara/algoritmos-%20portugu%EAs-%20cormen.pdf

Recursao Primitiva

adicao(x,y) =
y=0: id(x)
y>0: h(adicao(x,p(y)) ou seja, s(adicao(x,p(y))

adicao(3,2)=5
adicao(3,2)=s(adicao(3,1)) = s(s(adicao(3,0))) = s(s(id(3))=s(s(3))=s(4)=5

Recursao Primitiva
y=0: id(x)
adicao(x,y) = proj33(x, p(y), s(adicao(x,p(y)))

adicao(3,2)=proj33(3,1, s(adicao(3,1)))=proj33(3,1,s(4))=5
adicao(3,1)=proj33(3,0,s(adicao(3,0)))=proj33(3,0,s(3)))=4
adicao(3,0)=id(3)=3

Recursao While:
soma(x,y) = cond(y=0, id(x), soma(s(x), p(y))
sub(x,y) = cond(y=0, id(x), sub(p(x), p(y))

Recurso primitiva:
mult(x,y) = cond(y=0, id(y), soma(x, mult(x,p(y))))

//—————

zero(x) = cond(x = 0, p11(x), p22(p(x), zero(p(x))))
zero(2) = (2,1,0)=0
zero(2) = cond(x=0, p1,1(x), p2,2(p(2), zero(p(2)))
zero(1) = cond(x=0, p1,1(x) , p2,2(p(1), zero(p(1)))
zero(0) = cond(x=0, p1,1(x), p2,2 (p(0), zero(p(0)))

zero(x) = cond(x=0, 0, h(zero(p(x)))
zero(x) = cond(x = 0, p1(x), p2(p(x), zero(p(x))))

Recursiva primitiva:
mult(x,y)=cond(y=0, p11(zero(x)), p33(x, p(y), soma(x, mult(x, p(y))))

mult(x,y) = zero(y), para y=0

mult(x,y) = proj3,3(x, p(y), soma(x,mult(x,p(y)))

adicao(x,y) = proj33(x, p(y), s(adicao(x,p(y)))

mult(3,2) = proj33(3,1,soma(3, mult(3,1)))=proj33(3,1,soma(3,3))=6
mult(3,1)= proj33(3,0,soma(3, mult(3,0))=proj33(3,0,soma(3,0))=3
mult(3,0)= p11(zero(3))=p11(0)=0

fat(x) = cond(x=0, id(1), mult(x, fat(p(x))))

fat(2) = mult(2,fat(1)) = mult(1,1)=2
fat(1)= mult(1,fat(0)) = mult(1,1)=1
fat(0)= 1

Exemplo 8.3
Recursão Primitiva - Adição.

Suponha as seguintes funções:

id(x) = x função identidade $N \rightarrow N$
sucessor(x) = x + 1 função sucessor $N \rightarrow N$
proj33 (x,y,z) = z função projeção da 3ª componente da tripla $N^3 \rightarrow N$

A função adição nos naturais tal que: $N^2 \rightarrow N$
adição(x,y) = x + y
é definida usando recursão como segue:

adição (x, 0) = id(x)
adição (x, y + 1) = proj33 (x, y, sucessor(adição (x, y)))

——-

adicao(x,y)=
adição (x, 0) = id(x)
adição (x, y + 1) = proj33 (x, y, sucessor(adição (x, y)))
adicao(x,y) = proj33(x, p(y), s(adicao(x,p(y)))

adicao(3,2)= 3 + s(s(p(0)) = s(s(0))

Recursão While:

Exemplo 13 *(Função while recursiva)*

$$soma(x, y) = \begin{cases} x & \text{se } y = 0 \\ s(soma(x, p(y))) & \text{se } y > 0 \end{cases}$$

soma(3,1)=
s(soma(3,p(1)))=
s(soma(3,0))
s(3)=4

soma(3,2)=5
s(soma(3,p(2))=
s(soma(3,1))=s(4)=5
soma(3,1)=s(soma(3,p(1))= s(soma(3,0))=s(3)=4

Supondo que a função soma está definida.
mult(x,y)= zero(x)=0 se y = 0
y>0:
3 * 2 = 2 + 2 + 2 = 6
h(x,mult(x,p(y)))=
soma(x , mult(x , p(y)))=
mult(2,2)=
soma(2, mult(2, 1))= soma(2,2)=4
mult(2,1)= soma(2, mult(2,0))=soma(2,0)=2

I1: (0,1,2,2) = 2^0 * 3^1 * 5^2 * 7 ^2 = 3675
I2:(1,1,3,1) = 2^1 * 3^1 * 5^3 * 7 ^1=5250
I3: (0,2,4,4) = 2^0 * 3^2 * 5^4 * 7 ^4=13505625
I4:(1,2,1,5) = 2^1 * 3^2 * 5^1 * 7 ^5
I5: (0,3,6,6) = 2^0 * 3^3 * 5^6 * 7 ^6
I6:(1,2,0,5) = 2^1 * 3^2 * 5^0 * 7 ^5

c(I1,I2,I3,I4,I5,I6)=2 ^I1 * 3^I2 * 5^I3 * 7 ^I4 * 11^I5 * 13 ^I6
=2^3675 * 3 ^5250

Reposta: e)

Exercício 4.1

Analise o programa monolítico abaixo e assinale a opção que contém a codificação correta (considere a seguinte ordem de operações F=1, G=2, H=3 e dos testes T_1=1, T_2=2).

Programa Monolítico

```
1:    faça F vá-para 2
2:    se T1 então vá-para 3 senão vá-para 1
3:    faça G vá-para 4
4:    se T2 então vá-para 1 senão vá-para 5
5:    faça H vá-para 6
6:    se T2 então vá-para 0 senão vá-para 5
```

a) $2^{3675} \bullet 3^{5250} \bullet 5^{135025} \bullet 7^{151260} \bullet 11^{496371875} \bullet 13^{30256}$

b) $2^{375} \bullet 3^{525} \bullet 5^{1350625} \bullet 7^{151630} \bullet 11^{4963311875} \bullet 13^{3025}$

c) $2^{3675} \bullet 3^{5250} \bullet 5^{13505625} \bullet 7^{1512630} \bullet 11^{49633171875} \bullet 13^{302526}$

d) $2^{375} \bullet 3^{5250} \bullet 5^{1355625} \bullet 7^{151260} \bullet 11^{49633171875} \bullet 13^{30256}$

e) $2^{3675} \bullet 3^{5250} \bullet 5^{13505625} \bullet 7^{152630} \bullet 11^{4963171875} \bullet 13^{3526}$