

1,5

4) a) if((a+b)/2 == 0) {
a++; ✓
}

if(x.id == 0) {
return NULL;
} ✓

printf(...);
✓

x += a; ✓

if(x == 1000);

4) b)

```

A) while (l > 0) {
    down(MUTEX); // apenas um programa vai entrar por vez.
    if ((a + b) % 2 == 0) {
        a++;
        up(MUTEX); // é feito logo após alterar a variável global.
    } else {
        up(MUTEX); // pode ser feito antes para acelerar o paralelismo.
        b++; // pois b não é alterado pelo outro.
    }
}
}

```

```

B) *A. {
    while (true) {
        down(MUTEX);
        if (x.id == 0) {
            up(MUTEX); // não vai alterar a var X.
            return NULL;
        }
        printf(...);
        up(MUTEX);
    }
}

```

```

*B() {
    int a = 0;
    for(;;) {
        down(MUTEX);
        x += a;
        if (x == 1000) {
            up(MUTEX);
            break;
        }
    }
}

```

```

up(MUTEX);
a++;
}
return NULL;
}

```

15

6) Variáveis de condição: cheio e vazio

Variáveis compartilhadas: mElementos

, buffer

Constantes: NMAX (guarda o tamanho do buffer)

```
void put(Elemento e){
    if (mElementos == NMAX)
        wait(chio);
    mElementos++;
    insereNoBuffer(e);
    if (mElementos == NMAX) OK
        signal(vazio);
}
```

```
Elemento get(){
    if (mElementos == 0)
        wait(vazio);
    mElementos--;
    if (mElementos == 0) OK
        signal(chio);
    return removeDoBuffer();
}
```