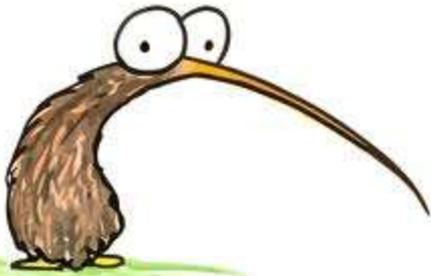


Java

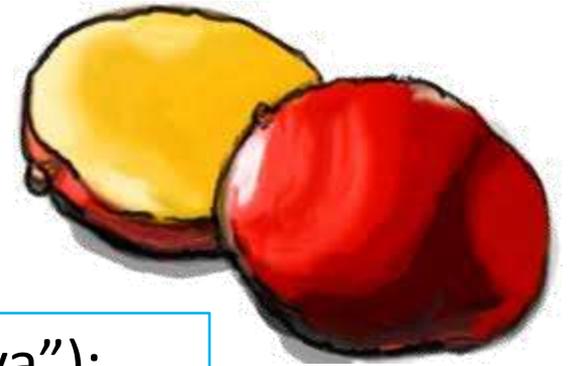
3

if, switch, while, for, do

Vitor Vaz da Silva

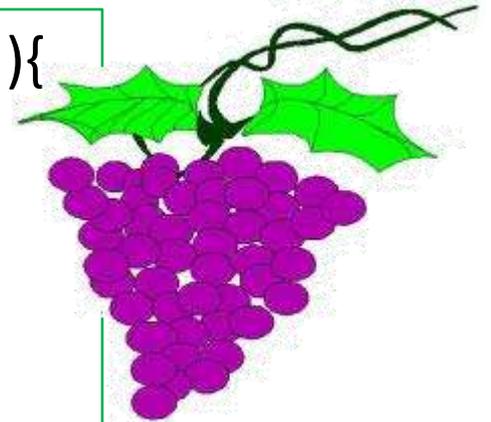


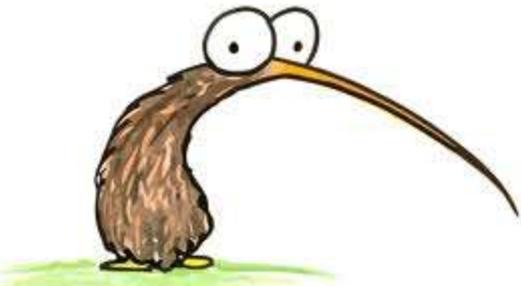
if



```
if (fruta != manga) escreve("pode ser maçã ou uva");  
else escreve ("acertou : é manga");
```

```
if ( caixaFruta.tem(manga) && caixaFruta.tem(uva) ){  
    escreve("Há manga e uva");  
    escreve ("e talvez haja maçã! ");  
}  
else{  
    escreve ("Talvez haja maçã! e ");  
    escreve ("Se houver uva não há manga,");  
    escreve ("Se houver manga não há uva,");  
    escreve ("e pode nem haver manga nem uva,");  
}
```





if



```
if (caixaFruta.quantidade(manga) > caixaFruta.quantidade(maca) ){  
    escreve("Há mais mangas do que maçãs");  
    escreve ("e nada sei acerca das uvas ");  
}  
else{  
    escreve ("Talvez haja mais maçãs do que mangas");  
    escreve ("ou tantas maçãs como mangas");  
    escreve ("Se houver uvas, não sei se há mais, menos ou o mesmo");  
    escreve ("que mangas ou maçãs");  
}
```





while

```
while( !boia.cheia() ){  
    boia.enche( bomba.ar() );  
    escreve("Continua...");  
}  
escreve ("Já encheu!");
```



```
while( jogador.tentativas() < jogo.tentativas() ){  
    escreve("Pode jogar");  
    jogador.joga();  
    jogador.actualiza( jogo.resultado() );  
}  
escreve ("Terminou!");
```



if while

**Na programação,
o if e o while
são suficientes
para resolver
todas as situações
do fluxo de informação**

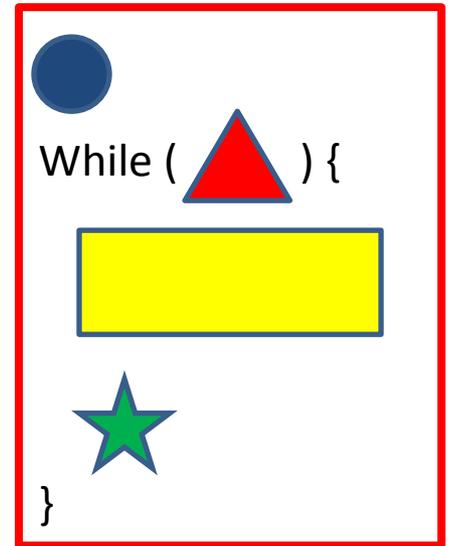


if while

- Mas há padrões que podem ser identificados e simplificados ... para que o programa fique mais legível, claro, pequeno, compreensível, e qualquer outra razão 😊



```
prepara_uma_vez
enquanto(isto){
    faz_coisas_diferentes_de_cada_vez
    no_fim_faz_sempre_o_mesmo
}
```



```
prep();

while(valor < resultado) {

    if(valor > 3) valor = resultado+1;
    if(resultado < 12) valor = valor-2;

    resultado = proximo();
}
```


switch

```
se (isto == for_igual_a_esta_coisa)
    faz_por_causa_disso
senão se (isto == for_igual_a_esta_coisa)
    faz_por_causa_disso
...
senão se (isto == for_igual_a_esta_coisa)
    faz_por_causa_disso
senão
    faz_qualquer_outra_coisa
```

```
ver_se (isto){
    for_igual_a_esta_coisa
        faz_por_causa_disso
    for_igual_a_esta_coisa
        faz_por_causa_disso
    ...
    senão_for_nada_disso
        faz_qualquer_outra_coisa
}
```

switch

```
if(num==1) s="um";  
else if(num==2) s="dois";  
else if(num==3) s="três";  
else if(num==5) s="cinco";  
else s="não é primo";
```

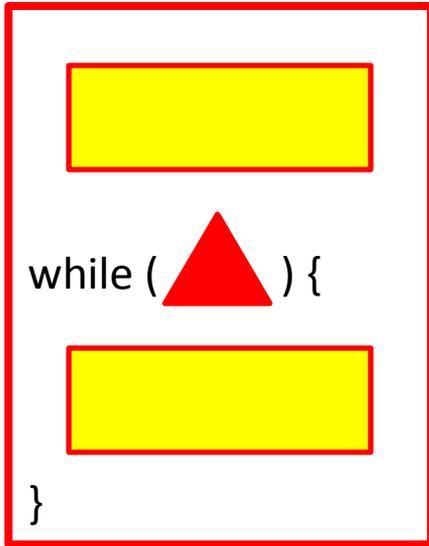


```
switch(num) {  
    case 1:  
        s="um";  
        break;  
    case 2:  
        s="dois";  
        break;  
    case 3:  
        s="três";  
        break;  
    case 5:  
        s="cinco";  
        break;  
    default:  
        s="não é primo";  
}
```

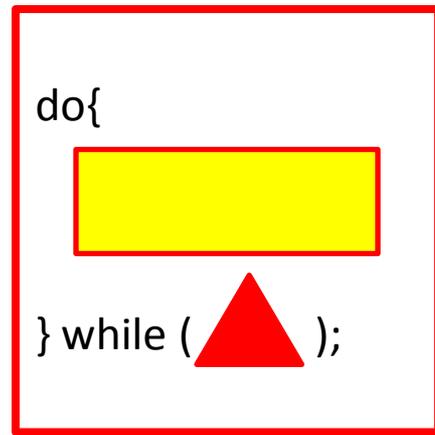
switch

```
switch(letra) {  
    case 'a':  
    case 'e':  
    case 'i':  
    case 'o':  
    case 'u':  
        res = "vogal";  
        break;  
    case 't':  
    case 'd':  
    case 'n':  
        tipo = "dental";  
    case 'f':  
    case 'v':  
        res = "consoante";  
}
```

do { ... } while



```
i=-1;  
do{  
    System.out.print("Letra: ");  
    arr[++i]=ler.next();  
}while(arr[i]!=null);
```

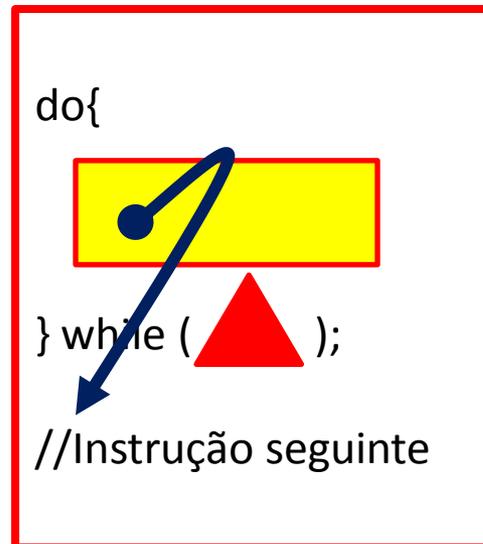
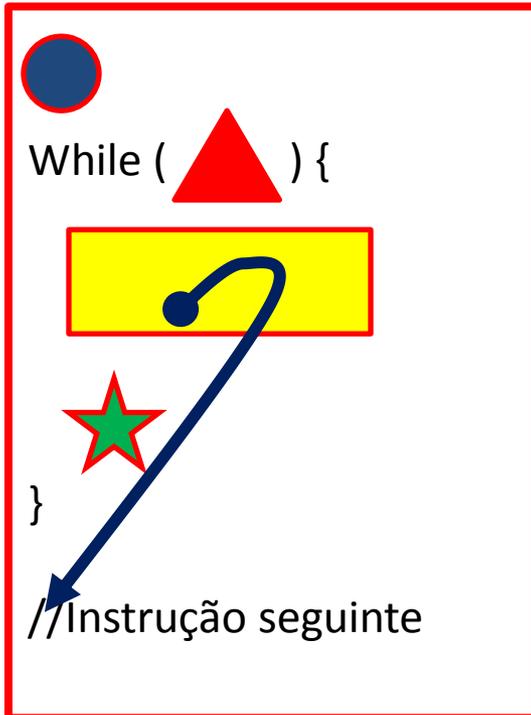
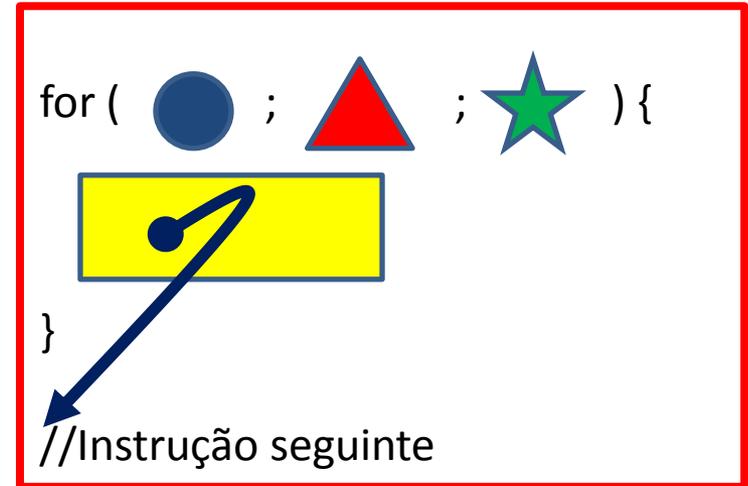


```
i=0;  
System.out.print("Letra: ");  
arr[i]=ler.next();  
while(arr[i]!=null){  
    System.out.print("Letra: ");  
    arr[i++]=ler.next();  
}
```

```
i=0;  
do{  
    System.out.print("Letra: ");  
    arr[i]=ler.next();  
}while(arr[i++]!=null);  
--i;
```

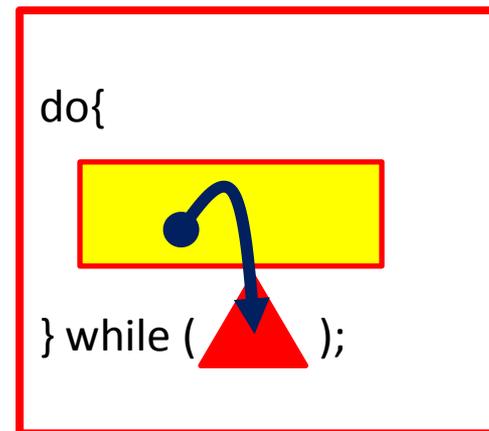
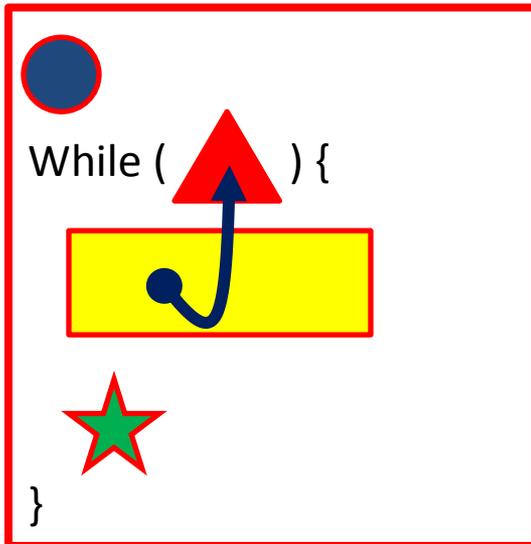
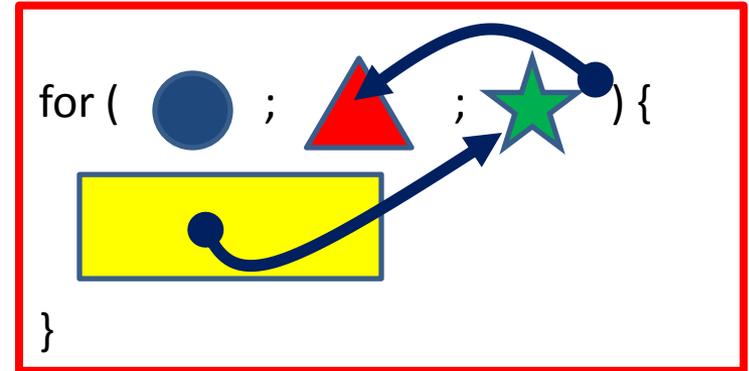
break

- break – quebra o fluxo
Sai do ciclo



continue

- Continue – verifica imediatamente a condição de permanência no ciclo. No caso do ciclo *for* o código do fim do ciclo é executado.



Expressão trenária

if() {



}

else {



}



?



:



;

```
if( a<b) {a++; b=3;} else {a=2; b--;} 
```

```
(a<b) ? (a++, b=3) : (a=2, b--);
```

Java
 C

```
if( a<b) {a=b+3;} else {a=2;} 
```

```
a = ( a<b) ? b+3 : 2;
```

Java
 C

```
s = ((n%2)!=0 ? "im" : "") + "par";
```

Java
 C

for - each

for(tipo variavel : objectoIteravel) blocoInstruções

```
int numeros[] = { 3, 4, 5, 8, 9, 10 };  
int soma = 0;
```

```
for(int idx=0; idx<numeros.length; idx++) soma+=numeros[idx];
```

```
for(int x : numeros) soma += x;
```

```
ArrayList<Double> lista = new ArrayList<Double>();
```

```
lista.add(1.99);
```

```
lista.add(3.19);
```

```
lista.add(5.20);
```

```
lista.add(4.99);
```

```
double soma = 0.0;
```

```
for(double valor : lista) soma+=valor;
```

Referências

- <http://tektonia.com>

