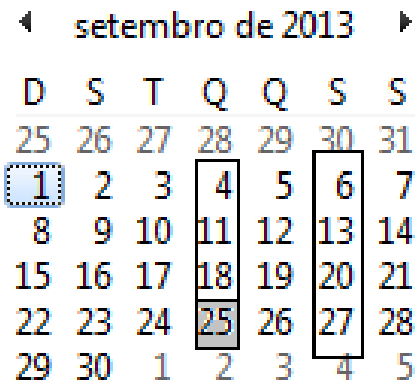
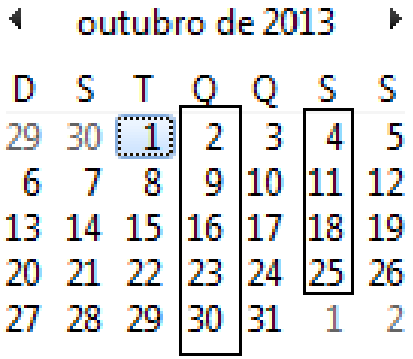
Incremento, decremento, atribuição
For – Repetição
Switch - Condição

Jose.wellington@uniceub.br

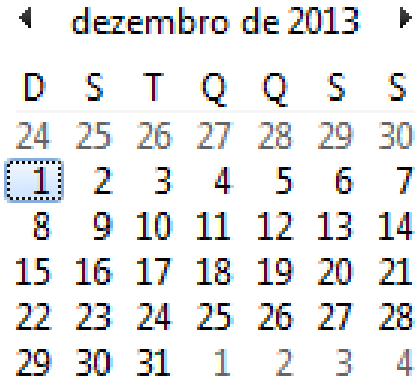
Calendário


agosto de 2013
D S T Q Q S S
28 29 30 31 1 2 3
4 5 6 7 8 9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31


setembro de 2013
D S T Q Q S S
25 26 27 28 29 30 31
1 2 3 4 5 6 7
8 9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 1 2 3 4 5


outubro de 2013
D S T Q Q S S
29 30 1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31 1 2


novembro de 2013
D S T Q Q S S
27 28 29 30 31 1 2
3 4 5 6 7 8 9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30


dezembro de 2013
D S T Q Q S S
24 25 26 27 28 29 30
1 2 3 4 5 6 7
8 9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31 1 2 3 4

Agenda

- Incremento
- Decremento
- Atribuição
- For – Repetição
- O comando break – continue
- Switch
- Exercício

Operações Lógicas

Operadores lógicos

- == -> igual a
- != -> diferente de
- > -> maior que
- < -> menor que
- >= -> maior ou igual a
- <= -> menor ou igual a

Operadores lógicos e de negação

Operadores lógicos

! -> não

&& -> operador lógico “e”

|| -> operador lógico “ou”

Operadores lógicos

O primeiro exemplo é representado por: (condicao_A **&&** condicao_B)

(condicao_A AND condicao_B)

Fazendo comparações

Exemplo é representado por: (condicao_A || condicao_B)

(condicao_A OR condicao_B)

Operadores Aritméticos

Operadores Aritméticos

Operador	*Uso*	*Descrição*
`+`	`op1 + op2`	Retorna a **soma** de `op1` e `op2`.
`-`	`op1 - op2`	Retorna a **subtração** de `op1` por `op2`.
`*`	`op1 * op2`	Retorna a **multiplicação** de `op1` por `op2`.
`/`	`op1 / op2`	Retorna a **divisão** de `op1` por `op2`.
`%`	`op1 % op2`	Retorna o **resto** da divisão de `op1` por `op2`.

```
if (count % 2 == 0)
```

Incremento - Decremento

a++ e a--

Usaremos muito, mas MUITO mesmo o incremento e o decremento de unidade:

a = a + 1;

a = a - 1;

Então, inventaram atalhos

a = a + 1 pode ser representado por a++ ou ++a

a = a - 1 pode ser representado por a-- ou --a

Diferença de a=++b e a=b++

1. a = b++

Mais uma vez, é um atalho em Java, uma forma mais simples de escrever as seguintes linhas:
a = b;
b++;

2. a = ++b
Analogamente, é o atalho que os programadores Java usam para representar as seguintes linhas de código:
++b;
a = b;

Atribuição

a = a + b, fazemos: a +=b

a = a - b, fazemos: a -=b

a = a * b, fazemos: a *=b

a = a / b, fazemos: a /=b

a = a % b, fazemos: a %=b

For - Repetição

For

- Geralmente, o que é possível fazer usando o for, é possível fazer usando o laço while em Java. Então, naturalmente vem a pergunta: por que e para que serve, então, o laço for?
- É o mesmo que perguntar 'por que usar a++' se podemos usar 'a=a+1'.
- Simples: por questão de simplicidade e praticidade.

For

- A sintaxe do laço for é a seguinte:

- for(valor ou condição inicial do contador; condição do laço; fator de mudança, decremento ou incremento)
- {códigos}

```
public class for1 {
    public static void main(String[] args) {
        for(int count=1 ; count <= 10 ; count++)
            {System.out.println(count);}
        }
    }
```

For

- **Exemplo 1: Contando até 10, com laço for**

```
public class for1_10 {
    public static void main(String[] args) {
        for(int count=1 ; count <= 10 ; count++)
            {System.out.println(count);}
        }
    }
```

For

- **Exemplo 2: contagem regressiva**

```
public class for10_1{
      public static void main(String[] args) {
            for(int count=10 ; count >= 1; count--){
                  System.out.println(count); }
       } }
```

For

- **O comando break**

- Break significa quebrar, parar, frear, interromper. E é isso que se faz.
  Quando o Java encontra esse comando pela frente, ele interrompe o laço/estrutura de controle ATUAL

For

- **O comando continue**

- Como o nome diz, ele 'continua' o laço. O comando break interrompe o laço, já o continue interrompe somente a iteração atual.

For

- **Exemplo Comando break e continue**

fazer um só teste: vamos checar se é múltiplo de 2. E se é múltiplo de 17 e 19, podemos parar a iteração.

```
public class continueTest {
        public static void main(String[] args) {

                for(int count=1 ; count <=1000000 ; count++){
                    if(count % 2 == 0){
                        System.out.println(count);
                        continue;           }
                    if((count % 17 == 0) && (count % 19 == 0)){
                        System.out.println(count);
                         break;          }                }
                } }
```

Exercício

Exercício 08

- Faça um programa que através do número de linhas e colunas imprima o numero de linhas e colunas definido com seu contador? 4 linhas e 6 colunas

```
1 2 3 4 5 6
1 2 3 4 5 6
1 2 3 4 5 6
1 2 3 4 5 6
```

Switch

Switch

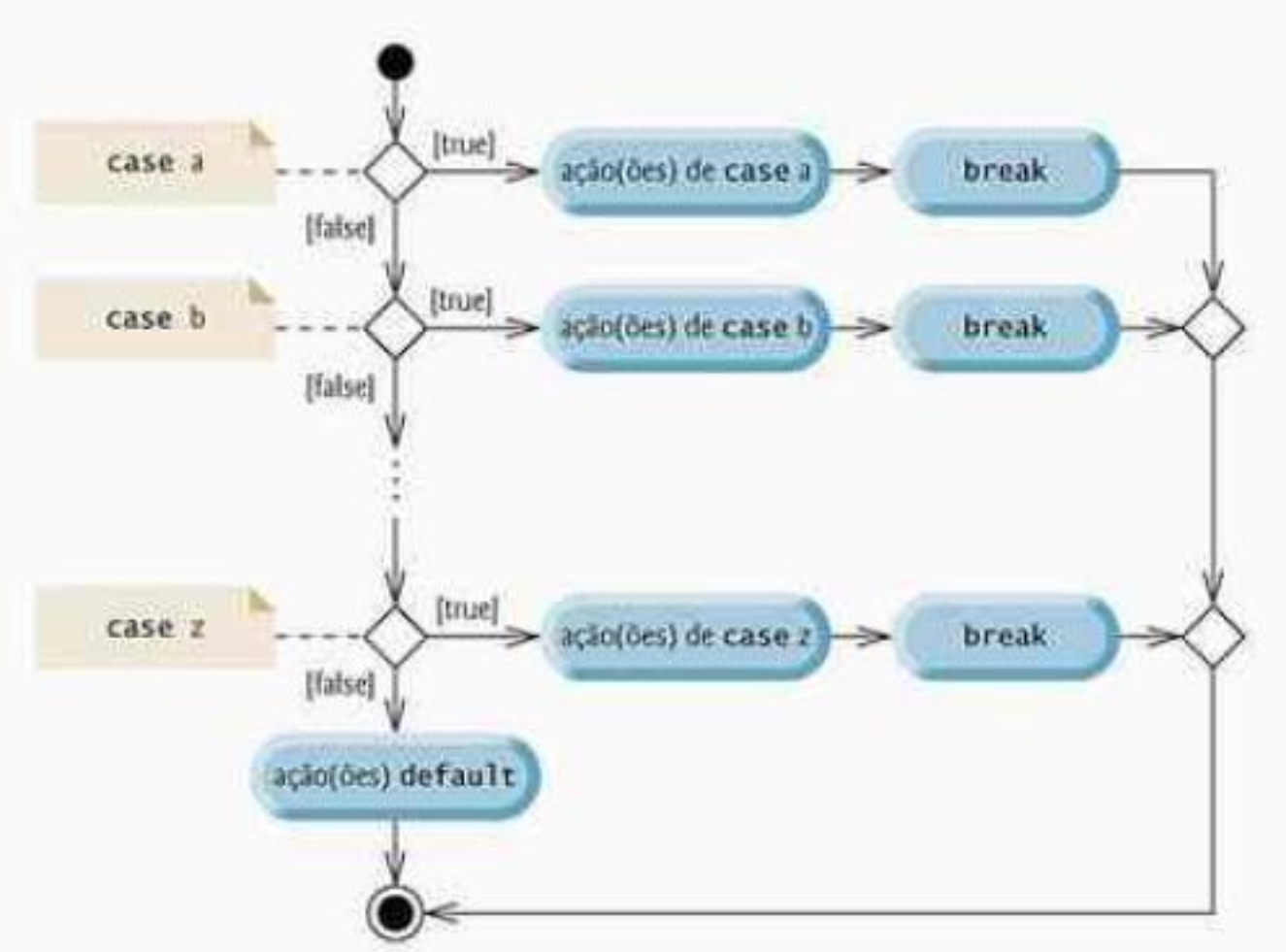
- Switch é uma declaração semelhante ao if, mas que usa valores inteiros para a tomada de decisões ao invés de expressões booleanas. (só pode ser usada em dados dos tipos short, int, byte ou char).

Switch
Estrutura de decisão múltipla

- **Usada quando precisamos escolher uma entre várias alternativas previamente definidas;**

```
switch(exp)
{
    case const1: comando1;
                 break;
    case const2: comando2;
                 break;
  ...
    case constn: comandon;
                 break;
    default: comando;
  }
```

Switch
Estrutura de decisão múltipla



Switch
Estrutura de decisão múltipla

```
01. public class ExemploSwitch {
02.     public static void main(String args[]) {
03.         int diaDaSemana = 1;
04.         switch (diaDaSemana) {
05.             case 1:
06.                 System.out.println("Domingo");
07.                 break;
08.             case 2:
09.                 System.out.println("Segunda-feira");
10.                 break;
11.             case 3:
12.                 System.out.println("Terça-feira");
13.                 break;
14.             case 4:
15.                 System.out.println("Quarta-feira");
16.                 break;
17.             case 5:
18.                 System.out.println("Quinta-feira");
19.                 break;
20.             case 6:
21.                 System.out.println("Sexta-feira");
22.                 break;
23.              case 7:
24.                 System.out.println("Sábado");
25.                 break;
26.             default:
27.                  System.out.println("Este não é um dia válido!");
28.         }
29.     }
30. }
```

Exercício

Exercício 08-01

- Atribua o número do mês e informe por extenso o mês.

Agenda

- Incremento
- Decremento
- Atribuição
- For – Repetição
- O comando break – continue
- Switch
- Exercício

- Jose.wellington@uniceub.br