

Programação 1

Aula 2

Valeri Skliarov, Prof. Catedrático

Email: skl@ua.pt

URL: <http://sweet.ua.pt/skl/>

Departamento de Eletrónica, Telecomunicações e Informática
Universidade de Aveiro

<http://elearning.ua.pt/>

Revisão da aula anterior

Pontos importantes

```
import java.util.*;
```

Importar a biblioteca de classes **Scanner**

```
Scanner sc = new Scanner(System.in);
```

Criar um objeto novo do tipo **Scanner**

Função **nextInt** do objeto do tipo **Scanner**

```
a = sc.nextInt();
```

Utilizar o objeto do tipo **Scanner**

Exemplo 1:

1. Importar Scanner

```
import java.util.*;
```

2. Criar objeto

```
Scanner objeto = new Scanner(System.in);
```

```
int inteiro;  
double real_v;
```

3. Usar objeto

```
System.out.print("Inteiro:");  
inteiro = objeto.nextInt();
```

```
System.out.print("Real:");  
real_v = objeto.nextDouble();
```

```
System.out.println("inteiro é " + inteiro + "; real é " + real);  
System.out.printf("inteiro é %d; real é %f\n",inteiro,real);
```

4. Fechar objeto

```
objeto.close();  
}  
}
```

```
Inteiro:2  
Real:34343  
inteiro e 2; real e 34343.0  
inteiro e 2; real e 34343.000000
```

Exemplo 2:

1. Importar Scanner

2. Criar objeto

3. Usar objeto

```
import java.util.*;
```

Agora o objeto deve ser estático

```
public class Nome {
```

```
    static Scanner objeto = new Scanner(System.in);
```

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

```
        int inteiro;
```

```
        double real_v;
```

```
        System.out.print("Inteiro:");
```

```
        inteiro = objeto.nextInt();
```

```
        System.out.print("Real:");
```

```
        real_v = objeto.nextDouble();
```

```
        System.out.println("inteiro é " + inteiro + "; real é " + real_v);
```

```
        System.out.printf("inteiro é %d; real é %f\n", inteiro, real_v);
```

```
    }
```

```
}
```

```
Inteiro:2
Real:34343
inteiro e 2; real e 34343.0
inteiro e 2; real e 34343.000000
```

Exemplo 3:

1. Importar Scanner

2. Criar objeto

3. Usar objeto

```
Welcome to DrJava. 1
> run Char
Char: A
Byte: 48
Char é A; Byte é 48
Char é A; Byte é 48
Char é A; Dec é 65
Hex é 41; Dec é 101
>
```

```
import java.util.*;
```

```
public class Nome {
```

```
    static Scanner objeto = new Scanner(System.in);
```

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

```
        char ch;
```

```
        byte by;
```

```
        System.out.print("Char: ");
```

```
        ch = objeto.nextLine().charAt(0);
```

```
        System.out.print("Byte: ");
```

```
        by = (byte) objeto.nextInt();
```

```
        System.out.println("Char é " + ch + "; Byte é " + by);
```

```
        System.out.printf("Char é %c; Byte é %d\n",ch,by);
```

```
        System.out.printf("Char é %c; Dec é %d\n",ch,(int)ch);
```

```
        System.out.printf("Hex é %h; Dec é %o\n",(int)ch,(int)ch);
```

```
    }
```

```
}
```

Agora o objeto deve ser estático

Exemplo 4:

Ok

```
import java.util.*;

public class Nome {
    static Scanner objeto = new Scanner(System.in);
    public static void main(String[] args) {
        int inteiro;
        double real;
        System.out.print("Inteiro:");
        inteiro = objeto.nextInt();
        real = Ler_real();

        System.out.println("inteiro é " + inteiro + "; real é " + real);
        System.out.printf("inteiro é %d; real é %f\n",inteiro,real);
    }

    public static double Ler_real()
    {
        double real;
        System.out.print("Real:");
        real = objeto.nextDouble();
        return real;
    }
}
```

Exemplo 5:

Erro

```
import java.util.*;
```

```
public class Nome {  
    public static void main(String[] args) {  
        Scanner objeto = new Scanner(System.in);
```

```
        int inteiro;
```

```
        double real;
```

```
        System.out.print("Inteiro:");
```

```
        inteiro = objeto.nextInt();
```

```
        real = Ler_real();
```

```
        System.out.println("inteiro é " + inteiro + "; real é " + real);
```

```
        System.out.printf("inteiro é %d; real é %f\n",inteiro,real);
```

```
    }
```

```
    public static double Ler_real()
```

```
    {
```

```
        double real;
```

```
        System.out.print("Real:");
```

```
        real = objeto.nextDouble();
```

```
        return real;
```

```
    }
```

```
}
```


Exemplo 6:

Ok

```
import java.util.*;
```

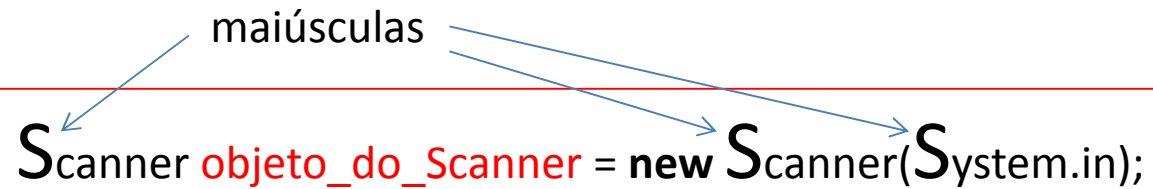
```
public class Nome {  
    public static void main(String[] args) {  
        Scanner objeto = new Scanner(System.in);  
        int inteiro;  
        double real;  
        System.out.print("Inteiro:");  
        inteiro = objeto.nextInt();  
        real = Ler_real();  
    }  
}
```

```
System.out.println("inteiro é " + inteiro + "; real é " + real);  
System.out.printf("inteiro é %d; real é %f\n",inteiro,real);  
objeto.close();
```

```
public static double Ler_real()  
{  
    Scanner objeto = new Scanner(System.in);  
    double real;  
    System.out.print("Real:");  
    real = objeto.nextDouble();  
    return real;  
}
```

Entrada de dados

Scanner objeto_do_Scanner = new Scanner(System.in);



int inteiro;

inteiro = objeto_do_Scanner.nextInt();

double real;

real = objeto_do_Scanner.nextDouble();



Saída de dados

double f; // Variáveis de entrada

// Ler dados

System.out.print("Double: ");

f = sc.nextDouble();

System.out.printf("Valor: %f",f);

System.out.println("\nValor: "+f+" "+f);

System.out.print("Última linha: ");

System.out.printf("\nValor: %10.3f",f);



Double: 1.2345678

Valor: 1.234568

Valor: 1.2345678 1.2345678

Última linha:

Valor: 1.235> |

10

```
public class saida_de_dados {  
    public static void main (String args[]) {  
        int a=3,b=4;  
        System.out.printf("boolean:  %b  %b\n",a==b,a!=b);  
        System.out.printf("char:    %c  %5c\n",a+0x30,b+0x30);  
        System.out.printf("char:%2c%4c\n",a+0x30+1,b+0x30+1);  
        System.out.printf("char:%2x%4x\n",a+0x30+1,b+0x30+1);  
    }  
}
```

```
boolean:    false    true  
char:      3        4  
char:  4      5  
char:34    35  
Press any key to continue . . .
```

Código ASCII

Diagram illustrating the ASCII code mapping across three columns: Código binário (Binary), Código octal (Octal), Código decimal (Decimal), and Código hexadecimal (Hexadecimal).

010 0001	041	33	21	!	100 0001	101	65	41	A	110 0001	141	97	61	a
010 0010	042	34	22	"	100 0010	102	66	42	B	110 0010	142	98	62	b
010 0011	043	35	23	#	100 0011	103	67	43	C	110 0011	143	99	63	c
010 0100	044	36	24	\$	100 0100	104	68	44	D	110 0100	144	100	64	d
010 0101	045	37	25	%	100 0101	105	69	45	E	110 0101	145	101	65	e
010 0110	046	38	26	&	100 0110	106	70	46	F	110 0110	146	102	66	f
010 0111	047	39	27	'	100 0111	107	71	47	G	110 0111	147	103	67	g
010 1000	050	40	28	(100 1000	110	72	48	H	110 1000	150	104	68	h
010 1001	051	41	29)	100 1001	111	73	49	I	110 1001	151	105	69	i
010 1010	052	42	2A	*	100 1010	112	74	4A	J	110 1010	152	106	6A	j
010 1011	053	43	2B	+	100 1011	113	75	4B	K	110 1011	153	107	6B	k
010 1100	054	44	2C	,	100 1100	114	76	4C	L	110 1100	154	108	6C	l
010 1101	055	45	2D	-	100 1101	115	77	4D	M	110 1101	155	109	6D	m
010 1110	056	46	2E	.	100 1110	116	78	4E	N	110 1110	156	110	6E	n
010 1111	057	47	2F	/	100 1111	117	79	4F	O	110 1111	157	111	6F	o
011 0000	060	48	30	0	101 0000	120	80	50	P	111 0000	160	112	70	p
011 0001	061	49	31	1	101 0001	121	81	51	Q	111 0001	161	113	71	q
011 0010	062	50	32	2	101 0010	122	82	52	R	111 0010	162	114	72	r
011 0011	063	51	33	3	101 0011	123	83	53	S	111 0011	163	115	73	s
011 0100	064	52	34	4	101 0100	124	84	54	T	111 0100	164	116	74	t
011 0101	065	53	35	5	101 0101	125	85	55	U	111 0101	165	117	75	u
011 0110	066	54	36	6	101 0110	126	86	56	V	111 0110	166	118	76	v
011 0111	067	55	37	7	101 0111	127	87	57	W	111 0111	167	119	77	w
011 1000	070	56	38	8	101 1000	130	88	58	X	111 1000	170	120	78	x
011 1001	071	57	39	9	101 1001	131	89	59	Y	111 1001	171	121	79	y
011 1010	072	58	3A	:	101 1010	132	90	5A	Z	111 1010	172	122	7A	z

Saída de dados formatada

```
public class saida_de_dados {  
    public static void main (String args[]) {  
        double a=3.123,b=4.5678;
```

```
1 System.out.printf("double:%7.2f%4.1f\n",a,b);  
2 System.out.printf("double:%07.2f%4.1f\n",a,b);  
3 System.out.printf("double:%07"+"%.2f%4.1f\n",a,b);  
4 System.out.printf("double:%07"+  
5   ".2f%4.1f\n",a,b);  
6 System.out.printf("double: %% %07.2f%4.1f\n",a,b);  
7 System.out.printf("double: %07.2f\t%4.1f\n",a,b);  
8 System.out.printf("double: %07.2f\t\t%4.1f\n",a,b);  
9 System.out.printf("double: %07.2f\b\b\b%4.1f\b\b\b \n",a,b);  
10 System.out.printf("double: %07.2f\b%4.1f\n",a,b);  
}
```

```
}
```

```
1 double: 3.12 4.6  
2 double:0003.12 4.6  
3 double:0003.12 4.6  
4 double:0003.12 4.6  
5 double:0003.12 4.6  
6 double: % 0003.12 4.6  
7 double: 0003.12 4.6  
8 double: 0003.12 4.6  
9 double: 0003 .6  
10 double: 0003.1 4.6  
Press any key to continue . . .
```

```
double: %07.2f  
double: 0003.12  
double: %07.2f\b\b\b  
double: 0003.12  
double: %07.2f\b\b\b%4.1f  
double: 0003 4.6  
double: %07.2f\b\b\b%4.1f\b\b\b  
double: 0003 4.6  
double: %07.2f\b\b\b%4.1f\b\b\b  
double: 0003 .6
```

back space (\b)

Mover cursor uma posição para trás

Identificadores

Identificadores – nomes utilizados para designar todos os objetos existentes num programa. Devem começar por uma letra ou por símbolo ‘_’ e só podem conter letras, números e símbolo ‘_’ (ex. nome, idade, i, j, cont_1, dia_mes, _km, res, nome1, n1_and_n3, a_____123_____b, t_, ...).

Exemplos errados: 3i, 1__nome, 22,
meu nome, o maior

Espaço não pode ser usado

Operadores - prioridades

Operators	Associativity
[] . () (method call)	Left
! ~ ++ -- + (unary) - (unary) () (cast) new	Right
* / % (modulus)	Left
+ -	Left
<< >> >>> (arithmetic shift)	Left
< > <= >= instanceof	Left
== !=	Left
& (bitwise and)	Left
^ (bitwise exclusive or)	Left
(bitwise or)	Left
&& (logical and)	Left
(logical or)	Left
? : (conditional)	Left
= += -= *= /= %= <<= >>= >>>= &= ^= =	Right

Operadores JAVA por prioridade decrescente

Exemplos:

```
public class if5 {  
    public static void main(String[] args) {  
        boolean A=true,B=false,C=false;  
        System.out.println((A&B) == (A&C));  
        System.out.println(A&B == A&C);  
    }  
}
```

```
true  
false  
Press any key to continue . . .
```

<code>== !=</code>	Left
<code>& (bitwise and)</code>	Left

```
public class if5 {  
    public static void main(String[] args) {  
        boolean A=true,B=false,C=false;  
        System.out.println(A == B == C);  
        System.out.println(A == B && B == C);  
        System.out.println((A == B) && (B == C));  
    }  
}
```

```
true  
false  
false  
Press any key to continue . . .
```


Erros potenciais

Quando vai copiar código no editor de Java a partir de outro editor podem aparecer erros na codificação de caracteres

Cuidado !!!

- Estruturas de controlo – decisão
- Tipos de dados **boolean**
- Operadores relacionais
- Operadores lógicos
- Estrutura de decisão **if**
- Estrutura de decisão múltipla **switch**

Alguns conceitos essenciais...

- Tipo de dados **boolean** (ou **Boolean**) – Podem assumir os valores **true** e **false** (verdadeiro e falso).
- Operadores relacionais: `<`, `<=`, `>`, `>=`, `==`, `!=`
- Operadores lógicos: `!`, `||`, `&&`
- Exemplos:
 - **boolean** cond1, cond2, cond3, cond4, cond5;
 - `cond1 = 3 > 0;` // cond1 fica com true
 - `cond2 = 5 != 5;` // cond2 fica com false
 - `cond3 = cond1 || cond2 = true || false;` // cond3
// fica com true
 - `cond4 = cond1 && cond2 = true && false;` // cond4
// fica com false
 - `cond5 = !cond4 = !false;` // cond5 fica com true

Instrução de decisão **if**

if (expressão) instrução;

- a expressão é avaliada;
- tem que ser uma expressão cujo resultado seja do tipo booleano;
- se verdadeira, é executada a instrução;
- se falsa, o programa continua na linha seguinte;

Exemplo 1:

```
if (_i__1__para_comparar >= _i__2__para_comparar)
    System.out.println("primeiro é maior ou igual a segundo");
else
    System.out.println("segundo é o maior");
```

```

import java.util.*;
public class bool
{
    public static void main (String args[])
    {
        Scanner sc = new Scanner(System.in);
        int A,B;
        boolean a,b;
        A = sc.nextInt(); B = sc.nextInt();
        if (A==B) System.out.println("A = B");
        else      System.out.println("A != B");
        if (A!=B) System.out.println("A != B");
        else      System.out.println("A = B");
        System.out.printf(A==B?"A = B":"A != B");
        System.out.println();
    }
}

```

Os resultados

```

3
6
A != B
A != B
A != B
Press any key to continue . . .

```

***Exemplo
(código
completo):***

```

import java.util.*;
public class bool {
    public static void main (String args[])
    {
        Scanner sc = new Scanner(System.in);
        int A,B,C,D;
        boolean y,a,b,c,d,e,f;
        A = sc.nextInt(); B = sc.nextInt(); C = sc.nextInt(); D = sc.nextInt();
        a = A==B; // a é true se A for igual a B e a é false no caso opósito
        b = B==C;
        c = C==D;
        d = A==C;
        e = A==D;
        f = B==D;
        if (a && b && c) System.out.println("A = B = C = D");

        y = !a && !b && !c && !d && !e && !f;
        if (y) System.out.println("todas valores A, B, C, D são diferentes");
        if (a || b || c || d || e || f)
            System.out.println("pelo menos dois valores do conjunto {A, B, C, D} sao iguais");

        System.out.printf("a and b and c = %b\n",a && b && c);
        System.out.printf("not(not a or not b or not c) = %b\n",!(!a || !b || !c));
    }
}

```

```

1
2
1
5
pelo menos dois valores do conjunto {A, B, C, D} sao iguais
a and b and c = false
not(not a or not b or not c) = false
Press any key to continue . . . _

```

Os resultados

**Exemplo
(código
completo):**

Para imprimir valor booleano

Exemplo (código completo):

```
import java.util.*;
public class bool
{
    public static void main (String args[])
    {
        Scanner sc = new Scanner(System.in);
        int A,B,C,D;
        A = sc.nextInt();
        B = sc.nextInt();
        C = sc.nextInt();
        D = sc.nextInt();

        if ((A==B)!=(C==D))

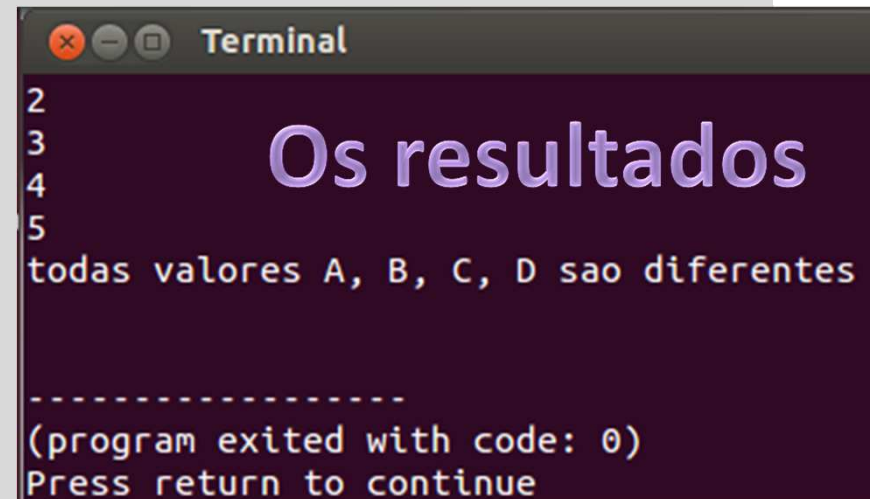
        if ((A>=B) && (B>=C) && (C>=D)) System.out.println("A >= B >= C >= D");

        if ((A!=B) && (A!=C) && (A!=D) && (B!=C) && (B!=D) && (C!=D))

    }
}
```

```
System.out.println("A = B ou C = D" +
    " mas não pode ser A=B e C=D " );
```

```
System.out.println("todas valores A, B, " +
    "C, D são diferentes");
```

A terminal window titled "Terminal" with a dark background. It shows line numbers 2 through 5 on the left. The output text is "Os resultados" in large purple letters, followed by "todas valores A, B, C, D sao diferentes" in red. Below that is a dashed line, then "(program exited with code: 0)" and "Press return to continue" in red.

```
2
3      Os resultados
4
5      todas valores A, B, C, D sao diferentes
-----
(program exited with code: 0)
Press return to continue
```

Estruturas de controlo - decisão

- Uma das particularidades de um computador é a capacidade de repetir tarefas ou executar tarefas consoante determinadas condições.
- Para implementar programas mais complexos, temos a necessidade de executar instruções de forma condicional.
- Determinadas instruções só podem/devem ser executadas depois da avaliação de determinadas condições.
- As instruções que permitem condicionar a execução de outras designam-se por *estruturas de controlo*. Nestes slides vamos apresentar as *estruturas de decisão*.
- Temos em JAVA dois tipos de instruções de decisão: **if** e **switch**.

Instrução de decisão `if`

- `if` (expressão) instrução;
- a expressão é avaliada;
- tem que ser uma expressão cujo resultado seja do tipo booleano;
- se verdadeira, é executada a instrução;
- se falsa, o programa continua na linha seguinte;
- exemplo:
 - `int x;`
 - `System.out.print("Um valor inteiro:");`
 - `x = sc.nextInt();`
 - `if(x < 0) x = -x;`
 - `System.out.println("O valor absoluto é " + x);`

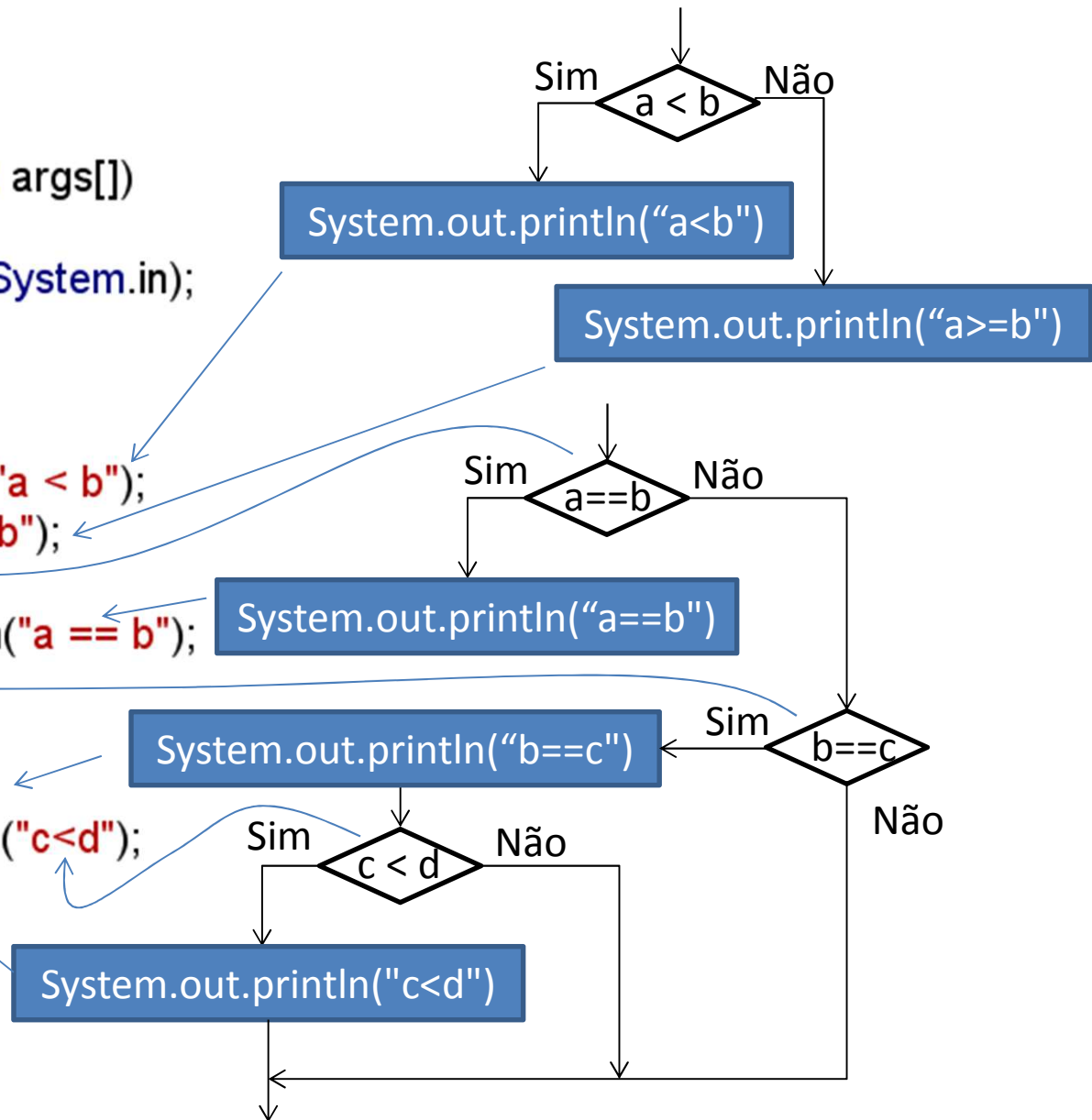
```
import java.util. *;
```

```
public class ex_if
{
    public static void main (String args[])
    {
        Scanner sc = new Scanner(System.in);

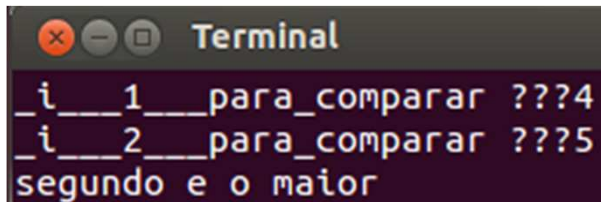
        int a=1, b=2, c=2, d=3;

        if (a < b) System.out.println("a < b");
        else System.out.println("a >= b");

        if (a == b) System.out.println("a == b");
        else if (b == c)
        {
            System.out.println("b == c");
            if (c < d) System.out.println("c < d");
        }
    }
}
```



Exemplo 1 (código completo):



A terminal window titled "Terminal" with a dark background. It displays the output of the Java program: the first prompt "_i__1__para_comparar ???4" is followed by the second prompt "_i__2__para_comparar ???5", and then the text "segundo e o maior" is printed.

```
_i__1__para_comparar ???4
_i__2__para_comparar ???5
segundo e o maior
```

```
import java.util.*;

public class if1 {
    public static void main(String[] args) {
        Scanner ob = new Scanner(System.in);

        int _i__1__para_comparar, _i__2__para_comparar;

        System.out.print("_i__1__para_comparar ???" );
        _i__1__para_comparar = ob.nextInt();

        System.out.print("_i__2__para_comparar ???" );
        _i__2__para_comparar = ob.nextInt();

        if (_i__1__para_comparar >= _i__2__para_comparar)
            System.out.println("primeiro é maior ou igual ao segundo");
        else
            System.out.println("segundo é maior");
        ob.close();
    }
}
```

Exemplo 2:

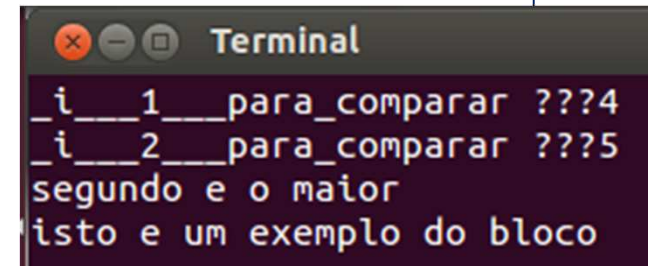
```
if (_i__1__para_comparar >= _i__2__para_comparar)
{
    System.out.println("primeiro é maior ou igual a segundo");
}
else
{
    System.out.println("segundo é o maior");
}
```

Pode usar chavetas {} mas estas não são necessários se quer executar uma só instrução

Um bloco é composto por mais que uma instrução

Exemplo 3:

```
if (_i__1__para_comparar >= _i__2__para_comparar)
{
    System.out.println("primeiro é maior ou igual ao segundo");
    System.out.println("este é o exemplo de bloco");
}
else
{
    System.out.println("segundo é maior");
    System.out.println("este é o exemplo de bloco");
}
```



```
Terminal
_i__1__para_comparar ???4
_i__2__para_comparar ???5
segundo e o maior
isto e um exemplo do bloco
```

Para blocos chavetas **{ }** são obrigatórias

Exemplo 4:

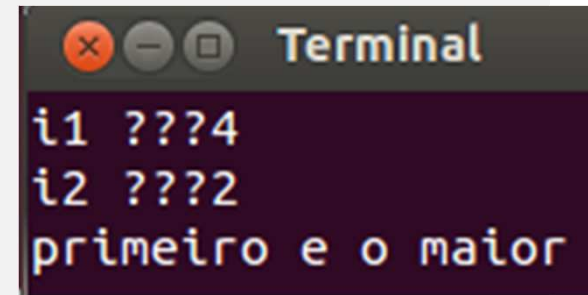
```
import java.util.*;

public class if2 {

    public static void main(String[] args) {
        Scanner ob = new Scanner(System.in);
        int i1;
        System.out.print("i1 ??? " );
        i1 = ob.nextInt();

        int i2;
        System.out.print("i2 ??? " );
        i2 = ob.nextInt();

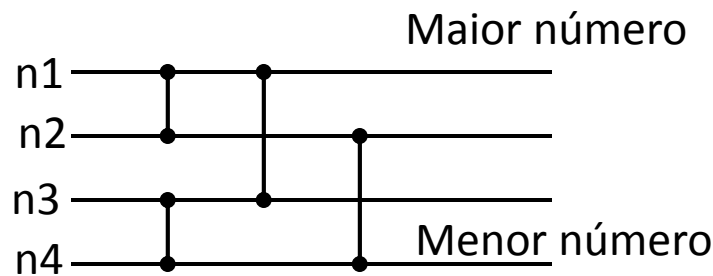
        if (i1 > i2)      System.out.println("primeiro é maior");
        else if (i1 == i2) System.out.println("primeiro é igual ao segundo");
        else             System.out.println("segundo é maior");
        ob.close();
    }
}
```

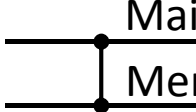
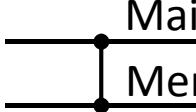


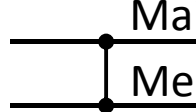
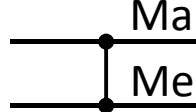
Exemplo 5: Pretende-se escrever um programa que dados quatro números inteiros introduzidos através do teclado imprime no terminal o maior e o menor número.

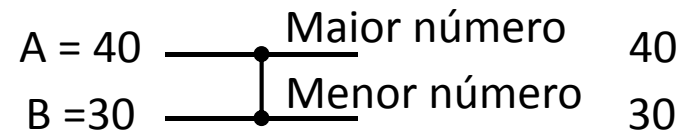
Escolha do próprio modelo de computação é muito importante !!!

Redes de procura

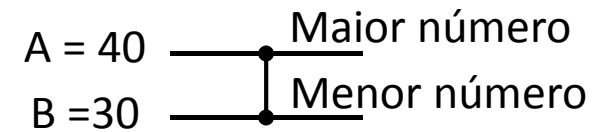


A = 10  Maior número 12
B = 12  Menor número 10

A = 40  Maior número 40
B = 30  Menor número 30



```
if (A < B) { tmp = A; A = B; B = tmp; }
```

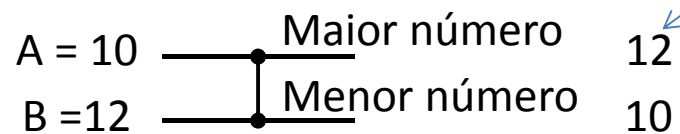


1 Um bloco : chavetas {} são obrigatórias

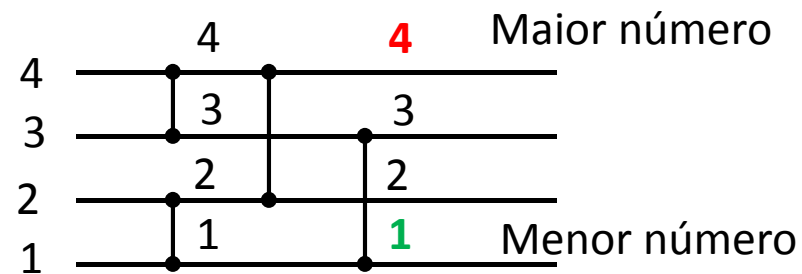
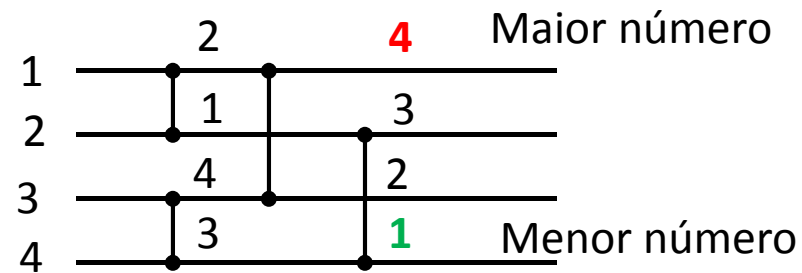
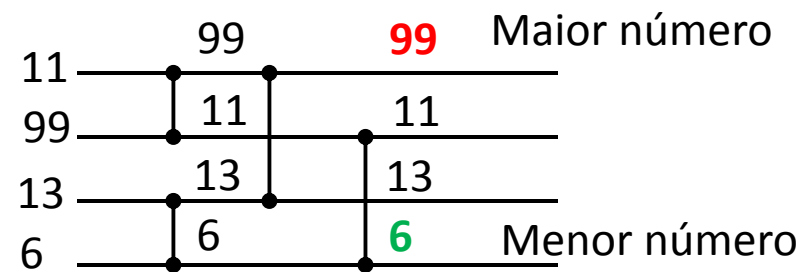
```
tmp=10
```

2
A=B=12

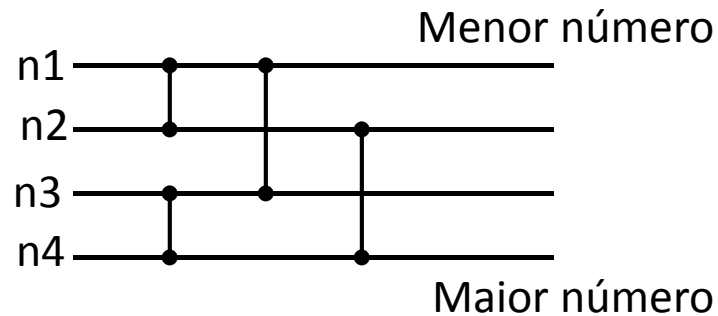
3
B=tmp=10



Exemplos:

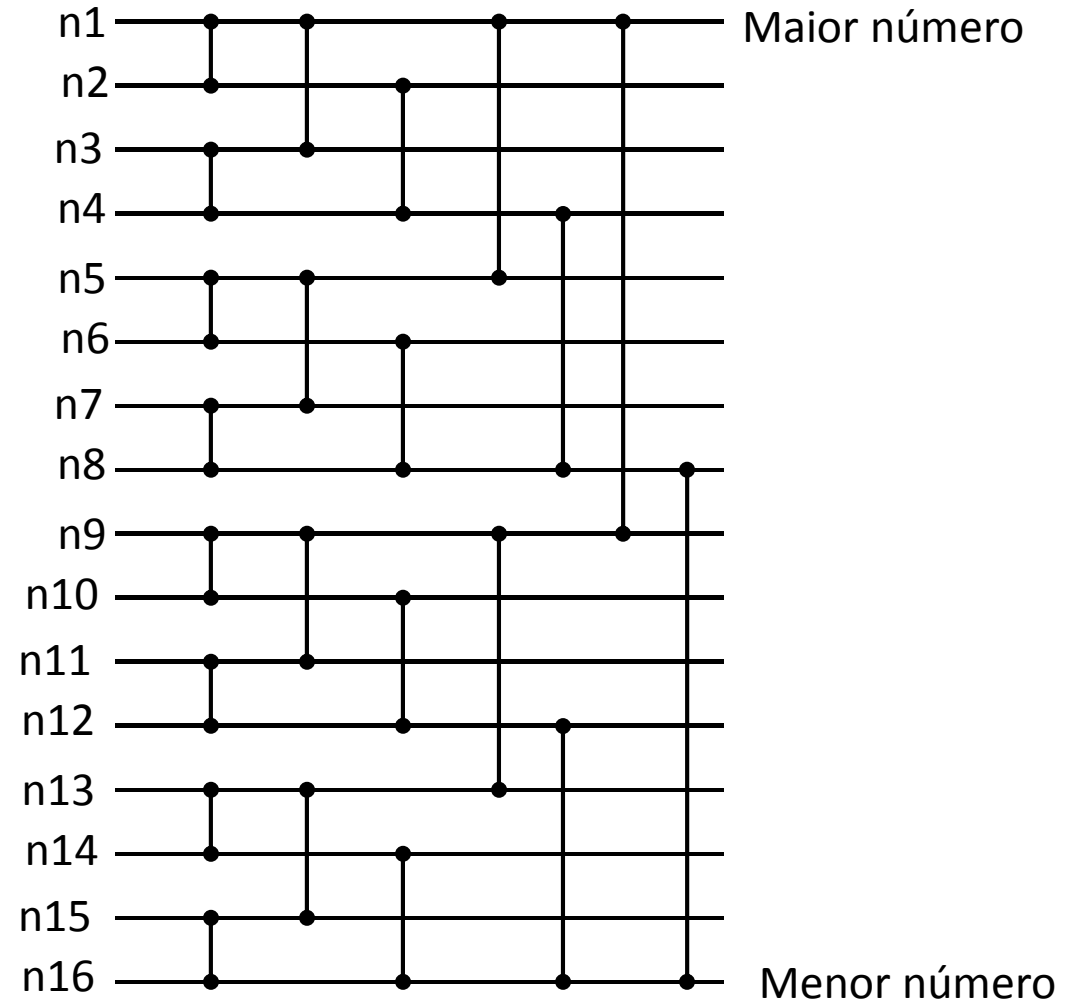
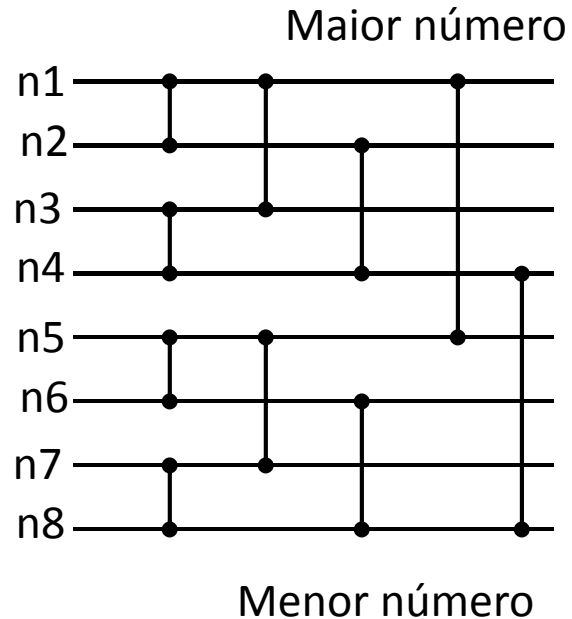


Exemplo 5:



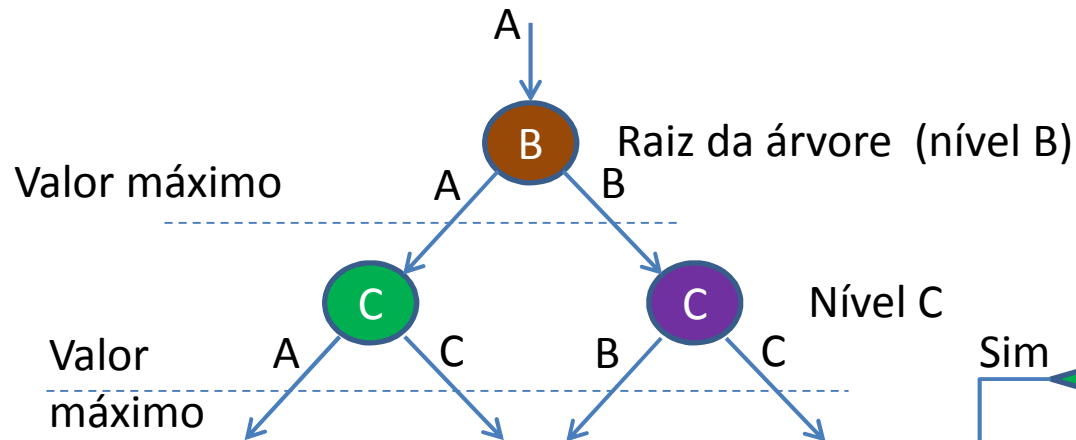
```
import java.util.*;
public class max_min {
    public static void main(String[] args) {
        Scanner ob = new Scanner(System.in);
        int n1,n2,n3,n4;
        System.out.print("n1 ???" );
        n1 = ob.nextInt();
        System.out.print("n1 ???" );
        n2 = ob.nextInt();
        System.out.print("n1 ???" );
        n3 = ob.nextInt();
        System.out.print("n1 ???" );
        n4 = ob.nextInt();
        int tmp;
        if (n1 > n2) { tmp = n1; n1 = n2; n2 = tmp; }
        if (n3 > n4) { tmp = n3; n3 = n4; n4 = tmp; }
        if (n1 > n3) { tmp = n1; n1 = n3; n3 = tmp; }
        System.out.println("O número menor é " + n1);
        if (n2 > n4) { tmp = n2; n2 = n4; n4 = tmp; }
        System.out.println("O número maior é " + n4);
        ob.close();
    }
}
```

Pretende-se escrever um programa que dados oito (dezasseis) números inteiros introduzidos através do teclado imprime no terminal o maior e o menor número.

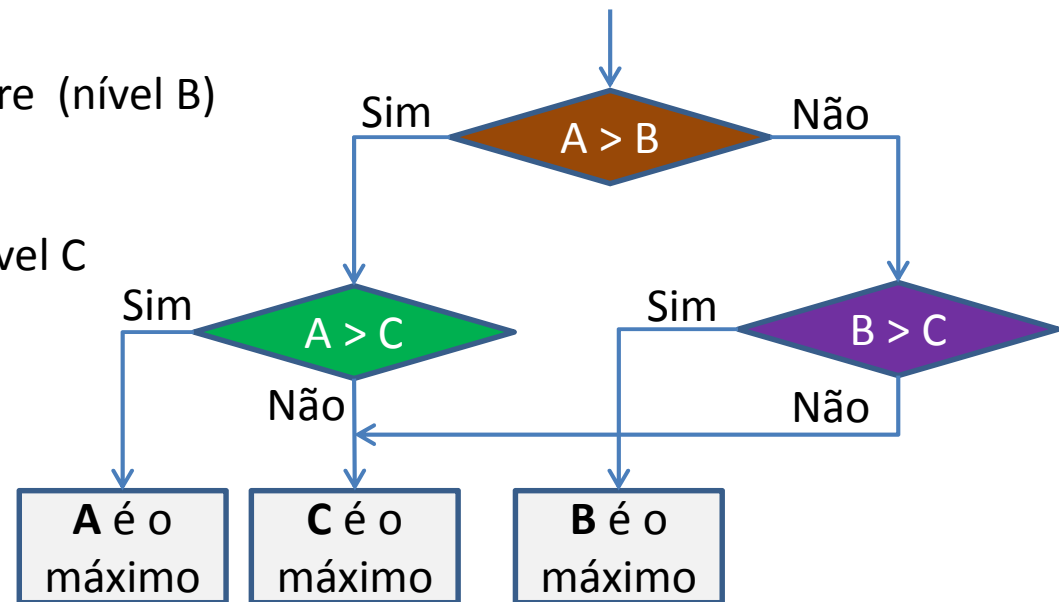


Árvores binárias

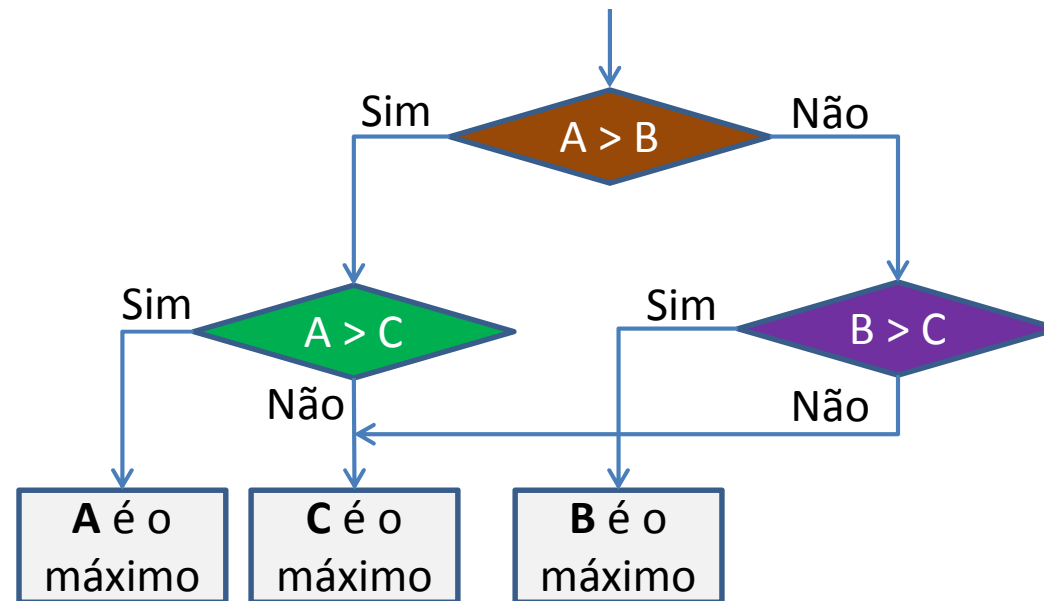
Pretende-se escrever um programa que dados três números inteiros (A, B, C) introduzidos através do teclado imprime no terminal o maior e o menor número.



Fluxograma (diagrama de blocos)

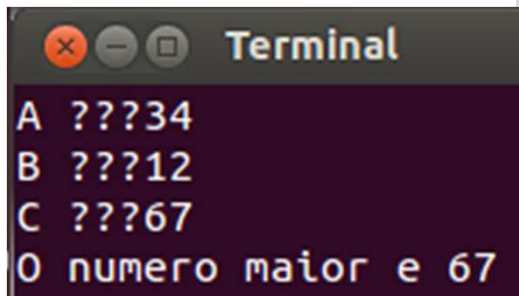


Pretende-se escrever um programa que dados três números inteiros (A, B, C) introduzidos através do teclado imprime no terminal o maior e o menor número.



```
if (A > B)
    if (A > C) System.out.println("O número maior é " + A);
    else      System.out.println("O número maior é " + C);
else
    if (B > C) System.out.println("O número maior é " + B);
    else      System.out.println("O número maior é " + C);
```

O código completo

A terminal window with a dark background and light-colored text. The title bar says "Terminal". The output shows four lines: "A ???34", "B ???12", "C ???67", and "O numero maior e 67".

```
Terminal
A ???34
B ???12
C ???67
O numero maior e 67
```

```
import java.util.*;
public class if3 {
    public static void main(String[] args) {
        Scanner ob = new Scanner(System.in);

        int A,B,C;

        System.out.print("A ???"   );
        A = ob.nextInt();
        System.out.print("B ???"   );
        B = ob.nextInt();
        System.out.print("C ???"   );
        C = ob.nextInt();

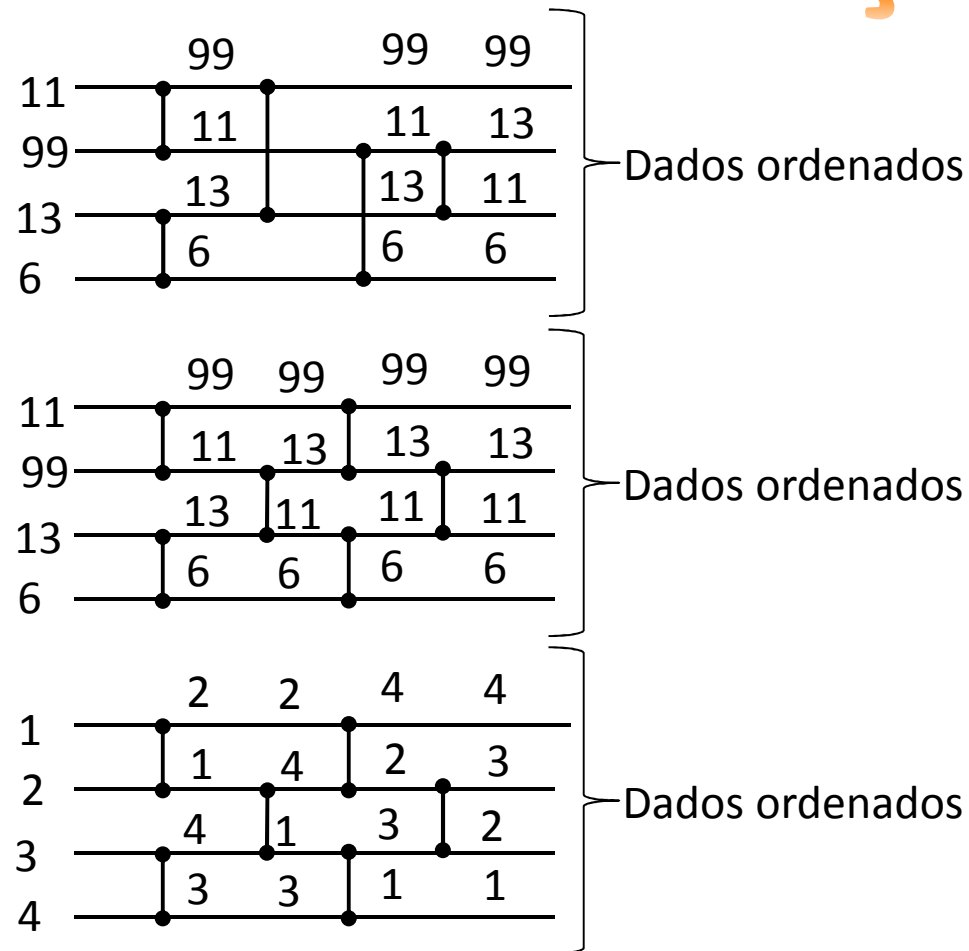
        if (A > B)
            if (A > C)    System.out.println("O número maior é " + A);
            else         System.out.println("O número maior é " + C);
        else
            if (B > C)    System.out.println("O número maior é " + B);
            else         System.out.println("O número maior é " + C);

        ob.close();
    }
}
```

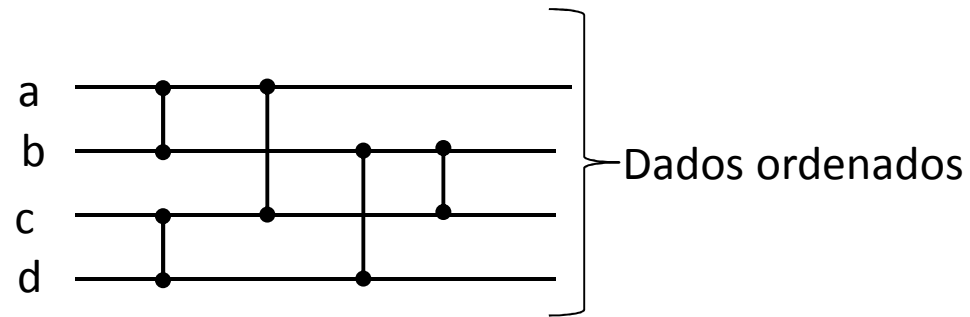
Exemplo 6: Pretende-se escrever um programa que dados quatro números inteiros introduzidos através do teclado imprime no terminal os números ordenados por ordem crescente.

Escolha do próprio modelo de computação é muito importante

Redes de ordenação

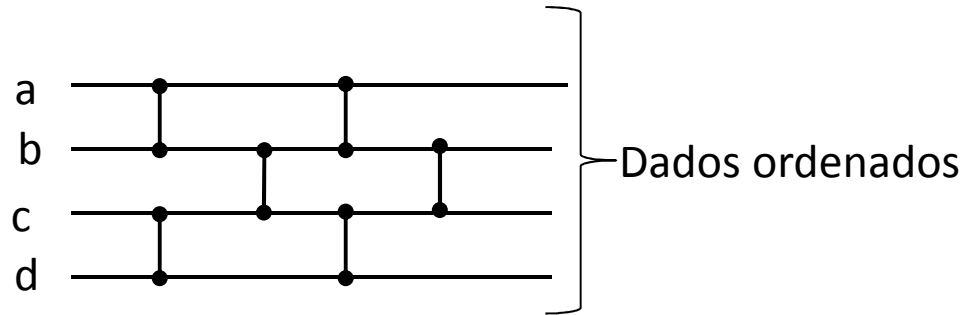


Exemplo 6:



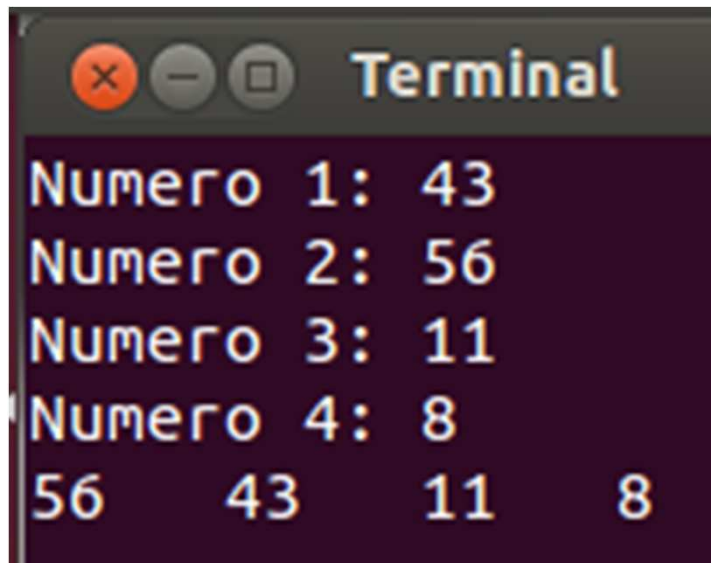
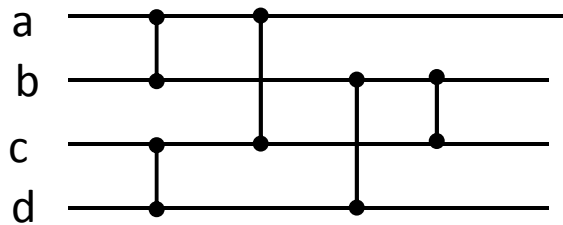
```
if(a > b) { tmp = a; a = b; b = tmp; } // rede de ordenção  
if(c > d) { tmp = c; c = d; d = tmp; }  
if(a > c) { tmp = c; c = a; a = tmp; }  
if(b > d) { tmp = b; b = d; d = tmp; }  
if(b > c) { tmp = c; c = b; b = tmp; }
```


Exemplo 6:



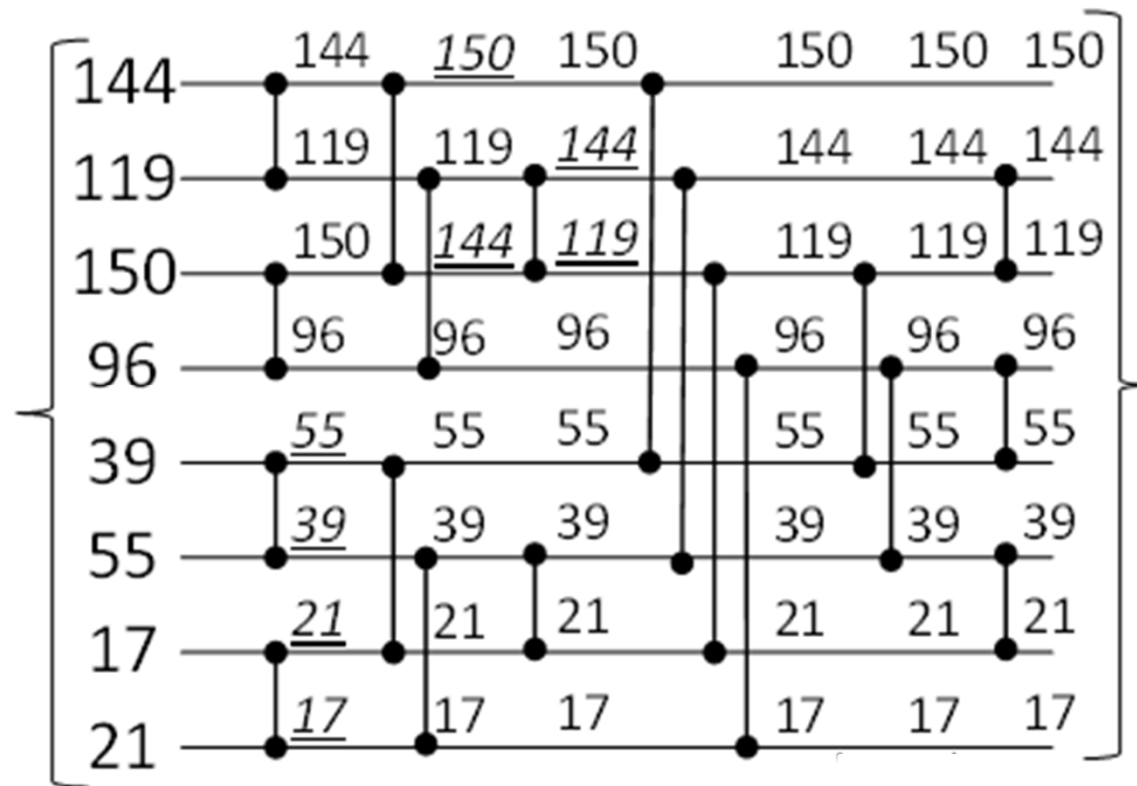
```
if(a > b) { tmp = a; a = b; b = tmp; } // rede de ordenção
if(c > d) { tmp = c; c = d; d = tmp; }
if(b > c) { tmp = c; c = b; b = tmp; }
if(a > b) { tmp = a; a = b; b = tmp; }
if(c > d) { tmp = c; c = d; d = tmp; }
if(b > c) { tmp = c; c = b; b = tmp; }
```

Exemplo 6:

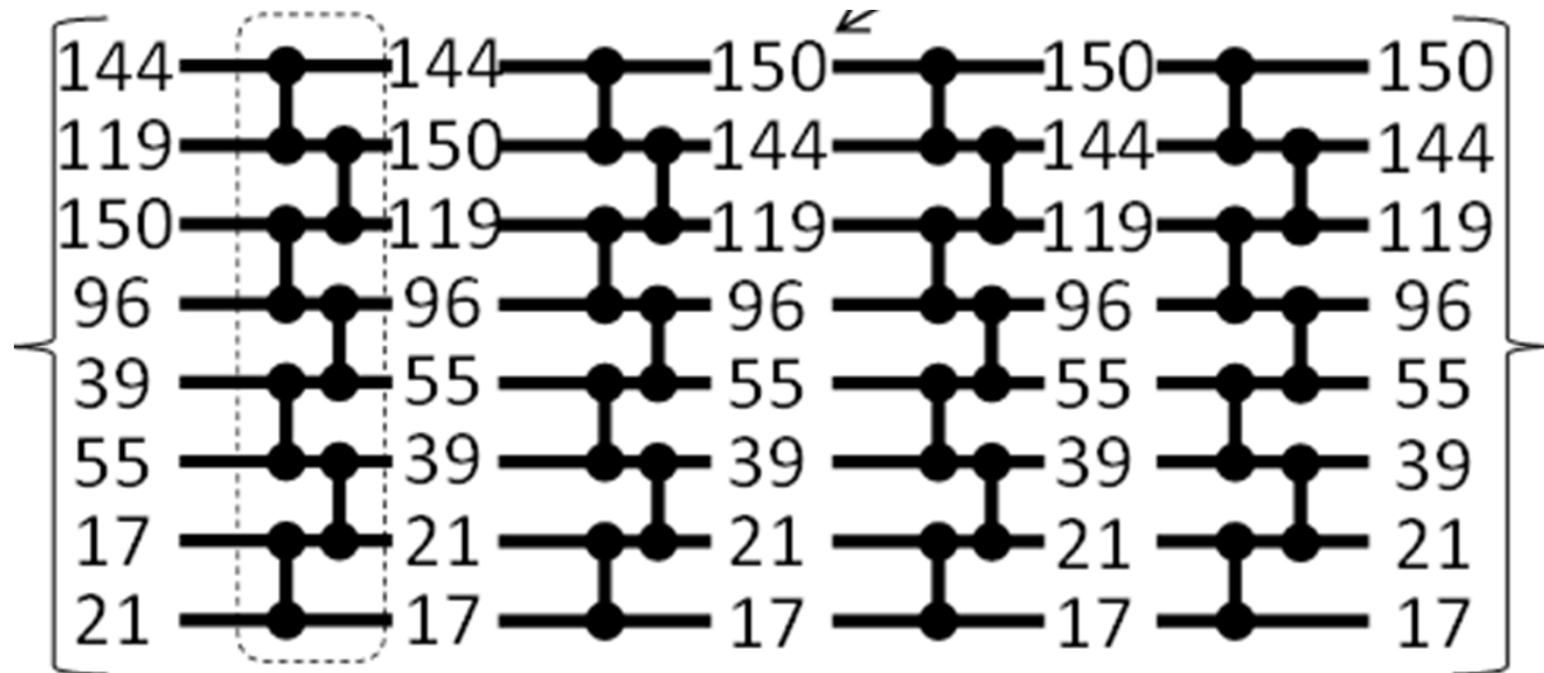


```
import java.util.*;
public class ex_0208_2014_4var
{
    public static void main (String args[])
    { Scanner sc = new Scanner(System.in);
      int a, b, c, d, tmp;
      System.out.print("Numero 1: ");
      a = sc.nextInt();
      System.out.print("Numero 2: ");
      b = sc.nextInt();
      System.out.print("Numero 3: ");
      c = sc.nextInt();
      System.out.print("Numero 4: ");
      d = sc.nextInt();
      if(a > b) { tmp = a; a = b; b = tmp; } // rede de ordenação
      if(c > d) { tmp = c; c = d; d = tmp; }
      if(a > c) { tmp = c; c = a; a = tmp; }
      if(b > d) { tmp = b; b = d; d = tmp; }
      if(b > c) { tmp = c; c = b; b = tmp; }
      System.out.printf("%d %d %d %d\n", d,c,b,a);
    }
}
```

Pretende-se escrever um programa que dados oito números inteiros introduzidos através do teclado imprime no terminal o maior e o menor número.



Pretende-se escrever um programa que dados oito números inteiros introduzidos através do teclado imprime no terminal o maior e o menor número.



Instrução de decisão múltipla `switch`

Algumas situações de decisão encadeadas com a instrução `if` podem ser resolvidas através da instrução de decisão múltipla **`switch`**.

- **`switch`** (expressão)
- {
- **`case`** valor1:
- bloco1;
- **`break`**;
- **`case`** valor2:
- bloco2;
- **`break`**;
- **`default`**:
- bloco3;
- }

- A expressão deve ser do tipo enumerado (número inteiro ou carater no caso dos tipos primitivos de JAVA – **`byte`, `short`, `int` ou `char`**).
- As constantes que constituem a lista de alternativas são do mesmo tipo da expressão.
- Primeiro é calculada a expressão e depois o seu valor é pesquisado na lista de alternativas existentes em cada **`case`**, pela ordem com que são especificados.
- Se a pesquisa for bem sucedida, o bloco de código correspondente é executado.
- Caso não exista na lista e se o **`default`** existir, o bloco de código correspondente é executado.
- A execução do **`switch`** só termina com o aparecimento da instrução **`break`**.

Exemplo 1:

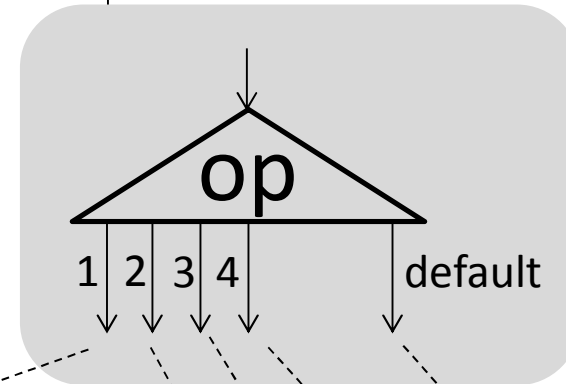
```
import java.util.*;

public class ex_switch
{
    public static void main (String args[])
    {
        Scanner sc = new Scanner(System.in);

        int a=2, b=3;
        int op;

        System.out.print("op ?");
        op = sc.nextInt();

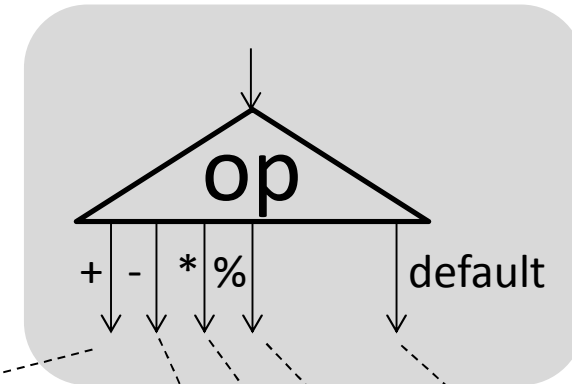
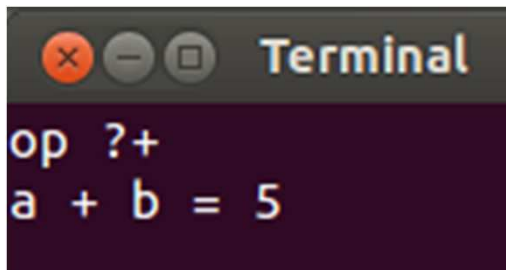
        switch(op)
        {
            case 1: <
                System.out.printf("a + b = %d\n", a+b);
                break; // remover break e experimentar
            case 2:
                System.out.printf("a - b = %d\n", a-b); <
                break;
            case 3:
                System.out.printf("a * b = %d\n", a*b); <
                break;
            case 4:
                System.out.printf("a %% b = %d\n", a%b); <
                break;
            default:
                System.out.println("Wrong operation"); <
        }
    }
}
```



```
Terminal
op ? 3
a * b = 6
```

Exemplo 2:

```
import java.util.*;
public class ex_switch_char
{
    public static void main (String args[])
    {
        Scanner sc = new Scanner(System.in);
        int a=2, b=3;   char op;
        System.out.print("op ?");
        op = sc.nextLine().charAt(0);
        switch(op)
        {
            case '+':
                System.out.printf("a + b = %d\n", a+b);
                break;
            case '-':
                System.out.printf("a - b = %d\n", a-b);
                break;
            case '*':
                System.out.printf("a * b = %d\n", a*b);
                break;
            case '%':
                System.out.printf("a %% b = %d\n", a%b);
                break;
            default:
                System.out.println("Operação errada");
        }
    }
}
```



Switch sem break

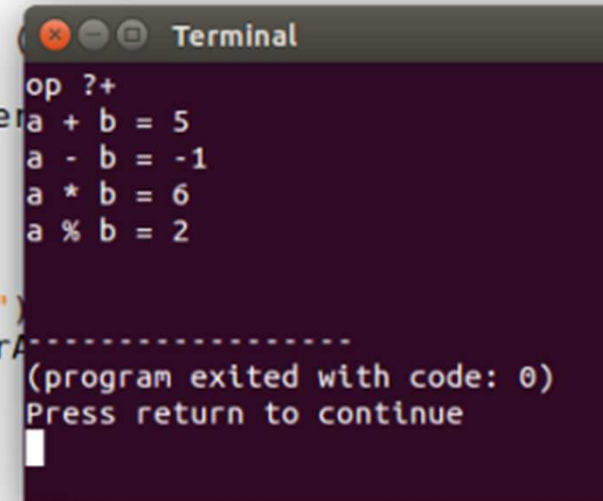
```
import java.util.*;

public class ex_switch_char
{
    public static void main (
    {
        Scanner sc = new Scanner

        int a=2, b=3;
        char op;

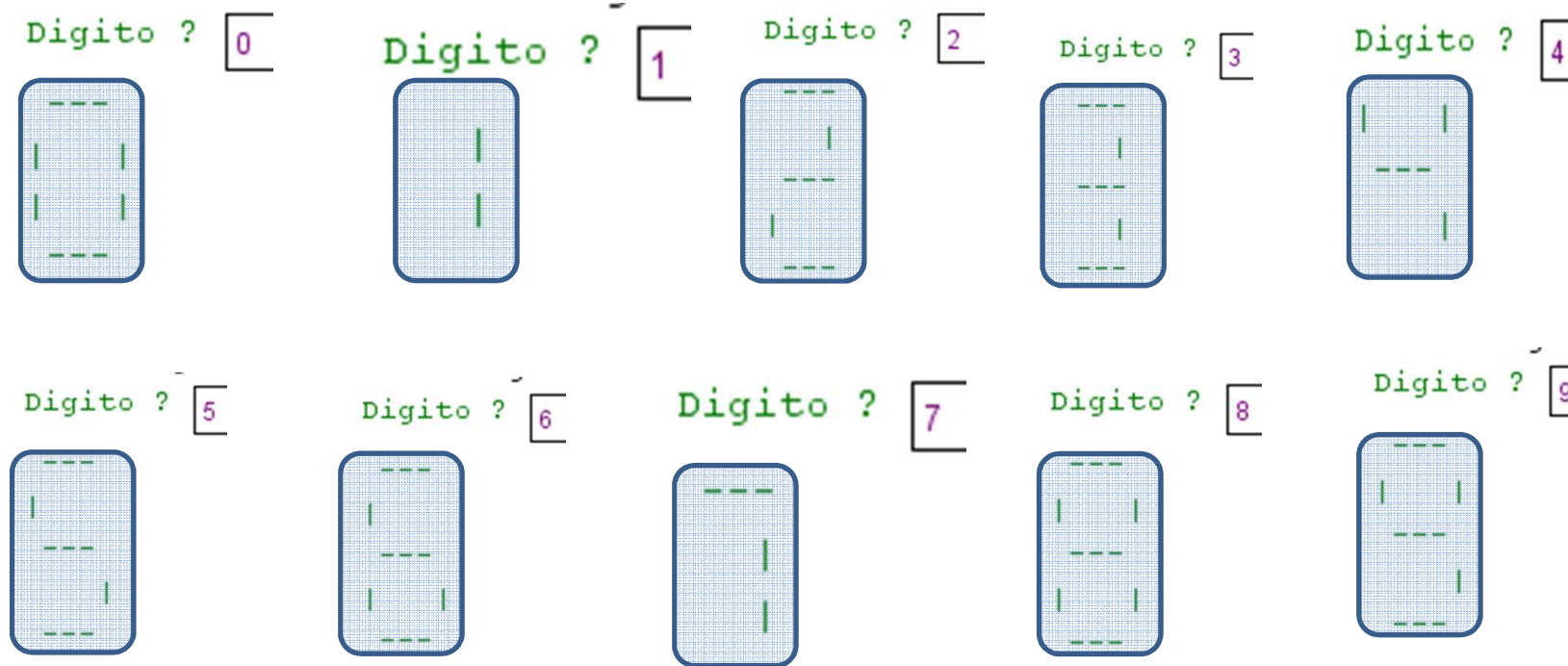
        System.out.print("op ?")
        op = sc.nextLine().charA

        switch(op)
        {
            case '+':
                System.out.printf("a + b = %d\n", a+b);
                // break;
            case '-':
                System.out.printf("a - b = %d\n", a-b);
                // break;
            case '*':
                System.out.printf("a * b = %d\n", a*b);
                // break;
            case '%':
                System.out.printf("a %% b = %d\n", a%b);
                break;
            default:
                System.out.println("Operacao errada");
        }
    }
}
```



Terminal window showing the output of the Java program. The prompt is 'op ?+'. The user has entered '+', and the program outputs 'a + b = 5'. The user has entered '-', and the program outputs 'a - b = -1'. The user has entered '*', and the program outputs 'a * b = 6'. The user has entered '%', and the program outputs 'a % b = 2'. The program then exits with the message '(program exited with code: 0) Press return to continue'.

Exemplo 3: Pretende-se escrever um programa que permite simular um controlador de display de segmentos de acordo com a figura abaixo.



Código binário



Código dos segmentos

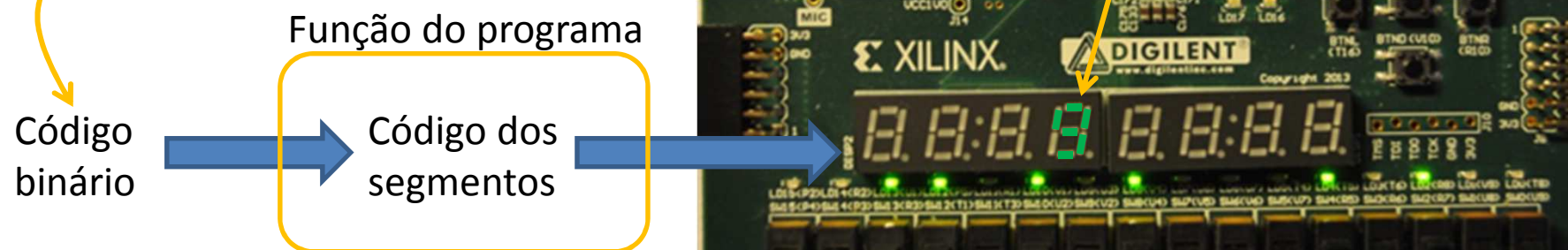


Exemplo 3: Pretende-se escrever um programa que permite simular um controlador de display de segmentos.

```
import java.util.*;
public class segment_control
{
    public static void main (String args[])
    {
        Scanner sc = new Scanner(System.in);
        int digito;
        System.out.print("Digito ?");
        digito = sc.nextInt();

        switch(digito) { case 0: case 2: case 3: case 5: case 6: case 7: case 8: case 9: System.out.println(" --- "); }
        switch(digito) { case 0: case 4: case 5: case 6: case 8: case 9: System.out.print("|"); }
        switch(digito) { case 5: case 6: System.out.println(); break;
                        case 0: case 1: case 2: case 3: case 4: case 7: case 8: case 9: System.out.println(" |"); }
        switch(digito) { case 2: case 3: case 4: case 5: case 6: case 8: case 9: System.out.println(" --- "); }
        switch(digito) { case 0: case 2: case 6: case 8: System.out.print("|"); }
        switch(digito) { case 2: System.out.println(); break;
                        case 0: case 1: case 3: case 6: case 7: case 8: System.out.println(" |"); break;
                        case 4: case 5: case 9: System.out.println(" |"); }
        switch(digito) { case 0: case 2: case 3: case 5: case 6: case 8: case 9: System.out.println(" --- "); }
    }
}
```

Exemplo 3: Pretende-se escrever um programa que permite simular um controlador de display de segmentos.



Exemplo 4: Pretende-se escrever um programa que permite verificar que um dado inteiro positivo é divisível por 2, 3 e 4 em simultâneo (resto de divisão é igual a 0).

```
import java.util.*;
public class Div
{ public static void main (String args[])
{ Scanner sc = new Scanner(System.in); int digito; System.out.print("Digito ?"); digito = sc.nextInt();

if (digito <= 0) { System.out.println("digito <= 0"); System.exit(1); }
switch(digito%2)
{ case 0:
    switch(digito%3)
    {
        case 0:
            switch(digito%4)
            {
                case 0: System.out.println("1: sim"); break;
                default: System.out.println("1: não");
            }
            break;
        default: System.out.println("1: não");
    }
    break;
    default: System.out.println("1: não");
}
}
```

Utilização de **switch**

Exemplo 4: Pretende-se escrever um programa que permite verificar que um dado inteiro positivo é divisível por 2, 3 e 4 em simultâneo (resto de divisão é igual a 0).

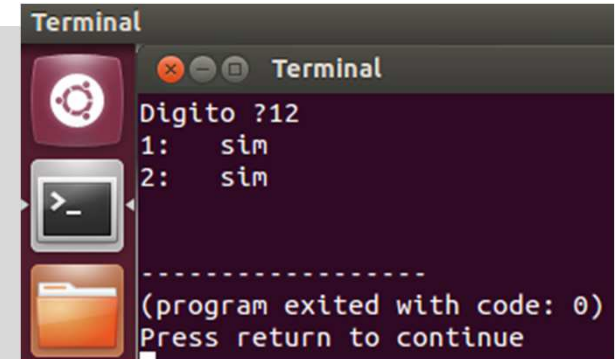
```
import java.util.*;
public class Div
{
    public static void main (String args[])
    {
        Scanner sc = new Scanner(System.in);
        int digito;
```

```
        System.out.print("Digito ?");
        digito = sc.nextInt();
```

```
        if (digito <= 0) { System.out.println("digito <= 0"); System.exit(1); }
```

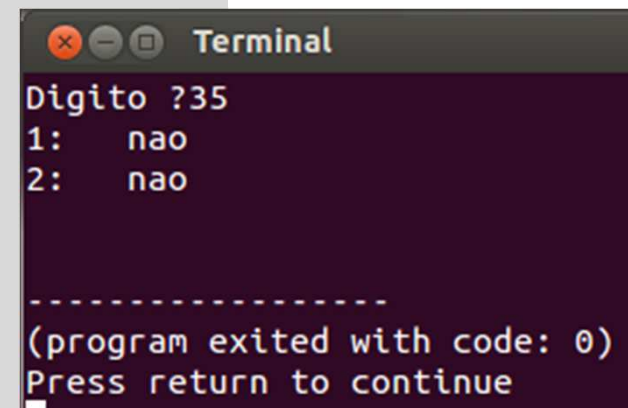
```
        if (digito%2 == 0)
            if (digito%3 == 0)
                if (digito%4 == 0) System.out.println("2: sim");
                else System.out.println("2: não");
            else System.out.println("2: não");
        else System.out.println("2: não");
    }
}
```

Utilização de if ... else



```
Terminal
Digito ?12
1:    sim
2:    sim

-----
(program exited with code: 0)
Press return to continue
```



```
Terminal
Digito ?35
1:    nao
2:    nao

-----
(program exited with code: 0)
Press return to continue
```

Pretende-se escrever um programa para calcular as raízes de uma equação de 2º grau do tipo $Ax^2+Bx+C = 0$, sendo os valores de A, B e C introduzidos do teclado. Tenha em atenção a possibilidade das soluções serem reais ou imaginárias.

A equação pode ter:

1. Duas soluções (**quando $B^2 - 4AC > 0$**);
2. Uma solução (neste caso podemos dizer que as duas soluções são iguais) (**quando $B^2 - 4AC = 0$**);
3. Nenhuma solução (neste caso podemos dizer que soluções são imaginárias) (**quando $B^2 - 4AC < 0$**);

$$X_{1,2} = (-B \pm \sqrt{B^2 - 4AC}) / 2a$$

ou

$$X_{1,2} = (-B/2 \pm \sqrt{(B/2)^2 - AC}) / a$$

```

import java.util.*;
public class ex_0211_2014
{ public static void main (String args[])
{
    Scanner sc = new Scanner(System.in);
    double A, B, C;
    double xPos = 0, xNeg = 0;
    int real_v;
    System.out.print("A: "); A = sc.nextDouble();
    System.out.print("B: "); B = sc.nextDouble();
    System.out.print("C: "); C = sc.nextDouble();

    if((Math.pow(B, 2) - 4 * A * C) > 0) real_v = 0;
    else if((Math.pow(B, 2) - 4 * A * C) == 0) real_v = 1;
    else real_v = 2;
    switch(real_v)
    {
        case 0:
            xPos = (- B + Math.sqrt(Math.pow(B, 2) - 4 * A * C)) / (2 * A);
            xNeg = (- B - Math.sqrt(Math.pow(B, 2) - 4 * A * C)) / (2 * A);
            System.out.printf("X = %.5f\nou\nX = %.5f\n",xPos, xNeg);
            break;

        case 1:
            xPos = (- B) / (2 * A);
            System.out.printf("X = %.5f\n", xPos);
            break;
    }
}
}

```

$$B^2 - 4AC > 0$$

$$B^2 - 4AC = 0$$

$$B^2 - 4AC < 0$$

case 2:

```
xPos = (- B) / (2 * A);
```

```
xNeg = Math.sqrt(4 * A * C - Math.pow(B, 2)) / (2 * A);
```

```
System.out.printf("X = %.5f + i %.5f\nou\nX = %.5f - i %.5f\n",xPos, xNeg, xPos, xNeg);
```

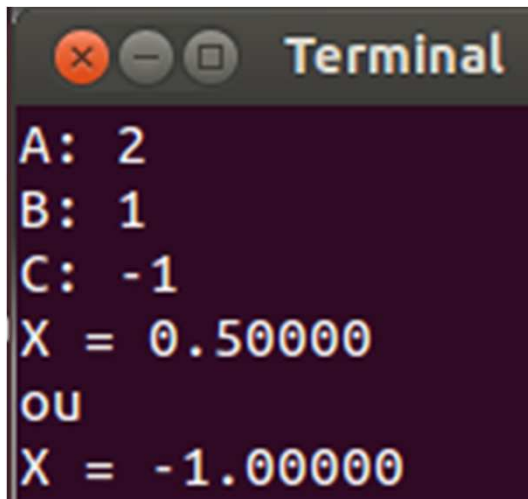
```
break;
```

```
default: System.out.println("ERRO");
```

```
}
```

```
}
```

```
}
```



```
Terminal
A: 2
B: 1
C: -1
X = 0.50000
ou
X = -1.00000
```

```
Welcome to DrJava.
```

```
> run ex_0211_2014
```

```
A: 
```

```
B: 
```

```
C: 
```

```
X = 3.00000
```

```
>
```

```
Welcome to DrJava. Working
```

```
> run ex_0211_2014
```

```
A: 
```

```
B: 
```

```
C: 
```

```
X = 0.16667 + i 1.28019
```

```
ou
```

```
X = 0.16667 - i 1.28019
```


Pretende-se escrever um programa que dada uma data composta pelo ano, considerando valores inteiros introduzidos através do teclado, calcula e escreve no terminal o número de dias em fevereiro. Um ano é bissexto de 4 em 4 anos, com exceção dos fins de século, que só são bissextos de 4 em 4 séculos.

```
import java.util.*;
```

```
public class February  
{
```

```
    public static void main (String args[])  
    {  
        Scanner sc = new Scanner(System.in);
```


```
        int year;
```

```
        System.out.print("Ano: ");  
        year = sc.nextInt();
```

```
        if( ( year % 4 == 0 ) && !(year % 100 == 0) ) || (year % 400 == 0) )  
            System.out.printf("O mes de fevereiro do ano %d tem 29 dias.\n", year);
```

```
        else
```

```
            System.out.printf("O mes de fevereiro do ano %d tem 28 dias.\n", year);  
        }  
    }
```

 Terminal

Ano: 1984

O mes de Fevereiro do ano 1984 tem 29 dias.

Operação ternária

```
import java.util.*;
public class Ternary_operation
{ public static void main(String args[])
{
    int A,B;
    Scanner read = new Scanner(System.in);
    System.out.print("A ? ");
    A = read.nextInt();
    System.out.print("B ? ");
    B = read.nextInt();
    System.out.print("max A, B = ");
    System.out.println(A > B ? A : B);

    int C = A > B ? A : B;
    System.out.println("max A, B = " + C);
}
}
```

> run Ternary_operation

A ?

45

B ?

12

max A, B = 45

max A, B = 45

```
import java.util.*;
```

```
public class LongLong {
```

```
    static Scanner objeto = new Scanner(System.in);
```

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

```
        byte by;
```

```
        System.out.print("Byte: ");
```

```
        by = (byte)objeto.nextInt();
```

```
        System.out.printf("Decimal: \t%3d\n",by);
```

```
        System.out.printf("Hexadecimal:\t%3h\n",by);
```

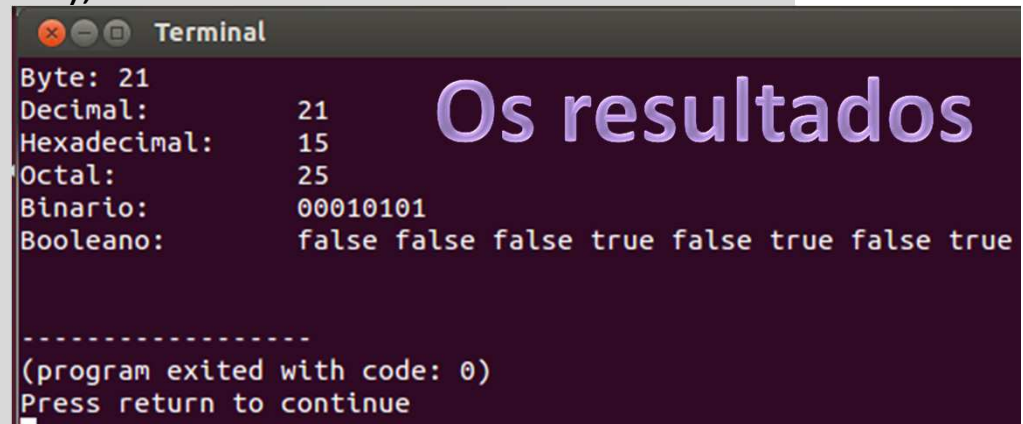
```
        System.out.printf("Octal:\t\t%3o\n",by);
```

```
        System.out.printf("Binário: \t %1h%1h%1h%1h%1h%1h%1h\n",  
                           0x1&by>>7,0x1&by>>6,0x1&by>>5,0x1&by>>4,  
                           0x1&by>>3,0x1&by>>2,0x1&by>>1,0x1&by);
```

```
        System.out.printf("Booleano: \t %b %b %b %b %b %b %b %b\n",  
                           (0!=(0x1&by>>7)),(0!=(0x1&by>>6)),  
                           (0!=(0x1&by>>5)),(0!=(0x1&by>>4)),  
                           (0!=(0x1&by>>3)),(0!=(0x1&by>>2)),  
                           (0!=(0x1&by>>1)),(0!=(0x1&by)));
```

```
    }
```

```
}
```



Terminal

```
Byte: 21  
Decimal:      21  
Hexadecimal:  15  
Octal:        25  
Binario:      00010101  
Booleano:     false false false true false true false true
```

Os resultados

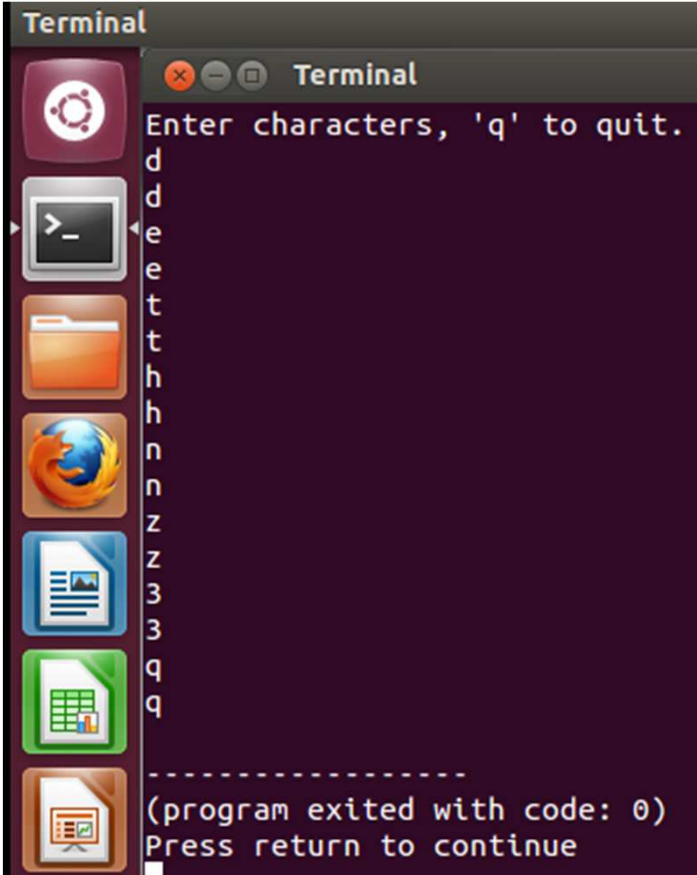
```
-----  
(program exited with code: 0)  
Press return to continue
```

```

import java.io.*;
public class BRRead
{ public static void main(String args[]) throws IOException
  {
    char c;
    BufferedReader br =new BufferedReader(new InputStreamReader(System.in));
    System.out.println("Enter characters, 'q' to quit.");
    do {
      c =(char)br.read();
      System.out.print(c); }
    while(c !='q');
  }
}

```

Um exemplo
mais complicado



```

Terminal
Enter characters, 'q' to quit.
d
d
e
e
t
t
h
h
n
n
z
z
3
3
q
q
-----
(program exited with code: 0)
Press return to continue

```

Conclusão

Escolha do próprio modelo de computação é muito importante !!!

Primeiro pensar depois escrever o código do programa !!!

Tentar descrever o algoritmo com um fluxograma !!!

Pensar sobre entrada, saída e formatação de entrada e saída !!!

Não esquecer inserir a linha `import java.util.*;` no início !!!

Não esquecer sobre a linha `Scanner sc = new Scanner(System.in);`
se necessário entrar os dados relevantes !!!

Use corretamente maiúsculas e minúsculas em nomes predefinidos