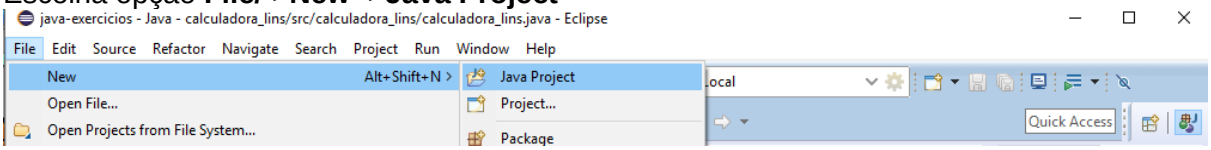
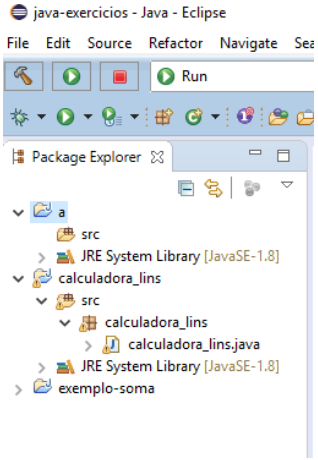
Introdução à programação em JAVA

1. Cria um novo projeto de Java.
Como fazer:

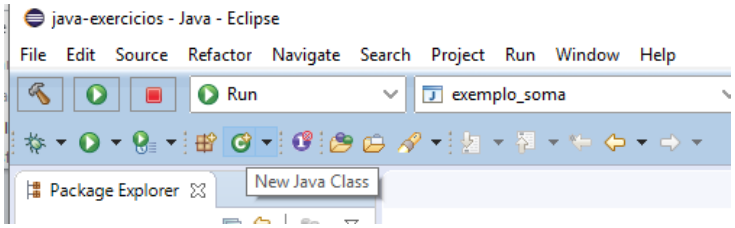
1. Escolha opção **File/->New ->Java Project**



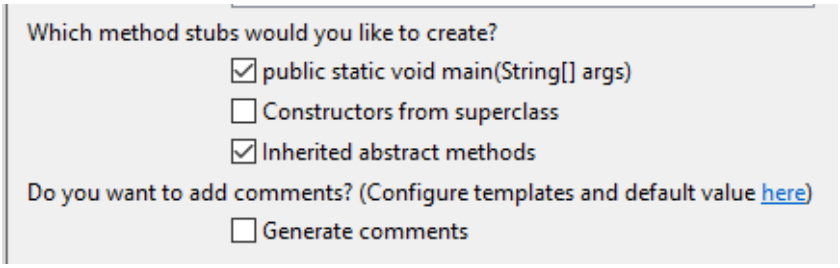
2. Na janela New Java Projet em:
   - **Project Name** colocar o nome do programa (por exemplo a)
   - **Location** criar uma pasta com o nome de um grupo ou manter o predefinido
   - Pressionar botão **Finish**



3. No Package Explorer aparece o ficheiro criado com extensão src
4. Adiconar uma nova classe para poder escrever o programa

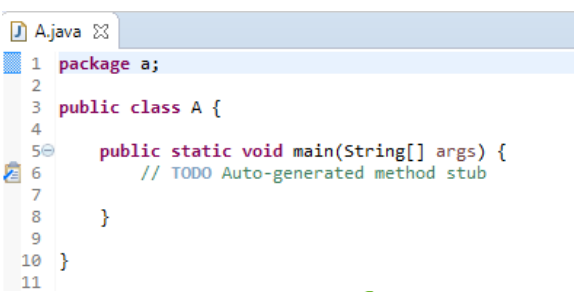


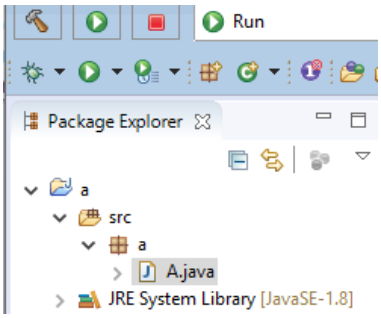
5. Na janela Java Class em:
   - Name colocar o nome da classe (por exemplo A –o nome das classes devem começar por maiúsculas)
   - Selecionar também o método public static void main…



   - Pressionar botão **Finish**

6. É criada o ficheiro A.java e deve-se escrever as instruções do programa dentro do método main





Métodos para leitura e escrita de dados:

System.out.println – escreve strings no ecrã

System é o nome de uma classe

- out é um objeto estático do tipo PrintStream membro de System

System. Out / System.err / System.in

- println é um método estático da classe PrintStream

System.in.read – Lê carateres do teclado (mais precisamente o seu código ASCII)

Estruturas de controlo

Estruturas condicionais: if-else, if-else-if, switch e default

If-else

Relembra - Permite indicar quais as circunstâncias em que determinada instrução ou conjunto de instruções deve ser executada.

Sintaxe:

```
If (condição)
  Instrução1;
[else Instrução2;]
```

Relembra - a componente else é facultativa (parêntesis retos).

A condição do if tem que estar sempre dentro de parêntesis. Tanto a instrução1 como a instrução2 (caso exista else) são seguidas de ponto e vírgula (;)

Funcionamento:

- Avalia uma condição;
- Se verdadeira executa a instrução imediatamente a seguir;
- Se falsa executa a instrução2 (caso exista o else).

Bloco de instruções:

Caso se deseje que um conjunto de instruções seja realizada quer no **if** quer no **else**, estas devem ser escritas entre **{ }** para que forme um único bloco*.

*bloco - conjunto formado por 2 ou mais instruções, delimitadas por { }.

Depois de um bloco não se coloca ponto e vírgula **(;)**.

Sempre que existam **if-else encadeados**, cada componente **else** pertence sempre ao último **if** (que ainda não tenha um **else** associado).

```
If (cond1)
     If (cond2)
          If(cond3)
               Inst1;
          else
               Inst2;
     else
          Inst3;
else
     inst4;
```

Exemplo1:

```
int idade = 15;
if (idade < 18) {
  System.out.println("Não pode entrar");
} else {
  System.out.println("Pode entrar");
}
```

Exemplo2:

```
int idade = 15;
 boolean amigoDoDono = true;
 if (idade >= 18 && amigoDoDono) {
   System.out.println("Pode entrar");
 }
 else {
   System.out.println("Não pode entrar");
 }
```

switch

A instrução switch adapta-se à tomada de decisões em que o n.º de hipóteses é elevado. Em geram mais do que duas (senão usamos a instrução if-else) de modo a simplificar a escrita do código (para não se tornar confuso e extenso).

Sintaxe:

```
switch (expressão)
{
   case  constante1 : instrução1;
   case  constante2 : instrução2;
        ...
   case  constanten : instruçãon;
   [ default : instruçãox;]
}
```

Funcionamento:

A expressão é qualquer uma cujo resultado seja um valor numérico do tipo char, int ou long.

A expressão é avaliada e em seguida o switch compara o resultado com o valor de cada constante, que segue cada um dos case:

Se o valor da expressão não for igual a nenhuma das constantes então são executadas as instruções que se seguem ao default (se existir).

- Se for igual a alguma das constantes que seguem os vários case, então são executadas todas as instruções que se seguem ao case correspondente. Para que não sejam executadas todas as instruções seguintes, necessitamos de usar outra instrução, que termina de imediato o switch.

break

- A instrução break permite parar a execução dentro de uma instrução switch, continuando o programa na instrução imediatamente a seguir ao switch.

Exemplo:

```
System.out.println("Concordas ou não?");
char c=(char)
System.in.read(); //*
  switch(c) {
case 's':
case 'S':
   System.out.println("Respondeste sim"); break;
case 'n':
case 'N':
   System.out.println("Respondeste não"); break;

 default: System.out.println("Resposta inválida");
 }
```

```
* método para leitura de dados tipo char-
correspondente à ordem na tabela asccii –
se introduzido s –corresponde a 115,
devendo ser feito o casting para char
```

Estruturas de Repetição: while, for, do...while. As instruções *break* e *continue* utilizadas em ciclos

while

A instrução while, executa uma instrução ou bloco de instruções enquanto uma determinada condição for verdadeira.

Sintaxe:

```
while (condição)
   Instrução;
```

Notas: Coloca-se entre parêntesis a condição que se tem que verificar para que a instrução ou bloco seja executado.

Funcionamento:

- Avalia uma condição;
- Se verdadeira executa a instrução ou bloco associado ao while;
- Se falsa o ciclo termina e o programa continua imediatamente a seguir ao while.

Exemplo:

```
int idade = 15;
while (idade < 18) {
  System.out.println(idade);
  idade = idade + 1;
}
```

for

A instrução for adapta-se à partida a situações em que se conhece o número de vezes que se repete o ciclo, isto é, o número de iterações do ciclo.

Sintaxe:

```
for ( inicializações ; condição ; pós-condição)
   Instrução;
```

Nota: O ciclo **for** identifica as suas componentes, separando-as por ponto e vírgula (;). Assim, se for necessário realizar mais do que uma inicialização ou mais do que uma pós-instrução, estas deverão ser separadas por vírgulas (,).

Funcionamento:

- É executado o código presente em inicializações (normalmente inicializações de variáveis presentes no ciclo). Esta componente é executada uma única vez!
- A condição é avaliada;
- Se for verdadeira, então é executada a instrução (ou bloco) do ciclo;
- Após a execução da instrução é executada a pós instrução. Nesta componente é realizado o incremento ou decremento da variável, etc., isto é, operações que permitam passar à próxima iteração do ciclo, voltando a condição a ser avaliada;
- Se for falsa (zero), então o **ciclo for** termina e o programa continua na instrução imediatamente a seguir;

Exemplo:

```
for (int i = 0; i < 10; i = i + 1)
{
  System.out.println("olá!");
}
```

do while

A instrução *do while*, difere dos ciclos anteriores porque o teste da condição é realizado no fim do corpo (instrução ou bloco de instruções) do ciclo e não antes, como acontecia com os ciclos **while** e **for**.

Desta forma o corpo do ciclo é executado pelo menos uma vez, enquanto que nos ciclos while e for o corpo do ciclo pode nunca ser executado (caso a condição seja falsa à partida).

Sintaxe:

```
do
   Instrução;
while (condição);
```

Funcionamento:

- A instrução (ou bloco de instruções) é executada;
- A condição é avaliada;
- Se verdadeira, volta a executar a instrução (ou bloco de instruções);
- Se falsa, o ciclo termina e o programa continua imediatamente a seguir ao while (condição).

Nota: O ciclo do…while adapta-se particularmente ao processamento de menus.

Ciclos (resumo)

	while	for	do … while

Sintaxe	while(cond)
	  Instrução;
		for(inic;cond;pos-cond)
		  Instrução;
			do
			  Instrução;
			while (cond);

Executa a instrução	Zero ou mais vezes	Zero ou mais vezes	Uma ou mais vezes
Testa a condição	Antes da instrução	Antes da instrução	Depois da instrução
Utilização	Frequente	Frequente	Pouco frequente

Break (num ciclo)

A instrução break, quando utilizada dentro dum ciclo, termina o correspondente ciclo, continuando a execução do programa na instrução imediatamente a seguir a esse ciclo.

Exemplo:

```
for (int i = x; i < y; i++) {
  if (i % 19 == 0) {
    System.out.println("Achei um número divisível por 19 entre x e y");
    break;
  }
}
```

continue (num ciclo)

A instrução continue, quando utilizada dentro dum ciclo, permite que a execução da instrução ou bloco de instruções corrente seja terminado, passando à próxima iteração do ciclo.

Exemplo:

```
for (int i = 0; i < 100; i++) {
  if (i > 50 && i < 60) {
    continue;
  }
  System.out.println(i);
}
```

EXERCÍCIOS

1. Descobre os erros.

```
int x = 10;                                          1
while (x > 0);
  System.out.println(x--);
System.out.println("We have lift off!");
```

```
int x = 10;                                          2
while (x > 0)
  System.out.println("x is " + x);
  x--;
```

```
int sum = 0;                                         3
for (; i < 10; sum += i++);
System.out.println("Sum is " + sum);
```

Para cada exercício, cria um novo arquivo com extensão **.java**, e declara aquele estranho cabeçalho, dando nome a uma classe (Exemplo: JAVA1_F2Ex1) que contenha um bloco main dentro dele:

```
public class JAVA1-F2Ex1 {
  public static void main(String[] args) {
    //o teu exercício é escrito aqui
  }
}
```

2. Escreve um programa que simule uma calculadora. Dados dois operandos constantes do tipo float efetue as operações de +,-,* e /.

3. Escreve um programa que imprima no ecrã o conjunto das 5 primeiras tabuadas (output semelhante a: 5*1= 5). Deve colocar uma linha em branco depois de cada tabuada.

4. Escreve um programa que:
   a) Imprima todos os números de 150 a 300.
   b) Imprima a soma de 1 até 1000.
   c) Imprima todos os múltiplos de 3, entre 1 e 100.

5. Escreve um programa que calcule o fatorial de 7, sabendo que:

   Fatorial de 7 = 7x6x5x4x3x2x1

6. Implementa o exemplo de utilização da função System.in.read .

```
...
int inChar;
    System.out.println("Insira um caráter:");
                    inChar = System.in.read();
                    System.out.print("Introduziu ");
                    System.out.println((char)inChar);
...
```

Alguns métodos úteis em dados do tipo String:

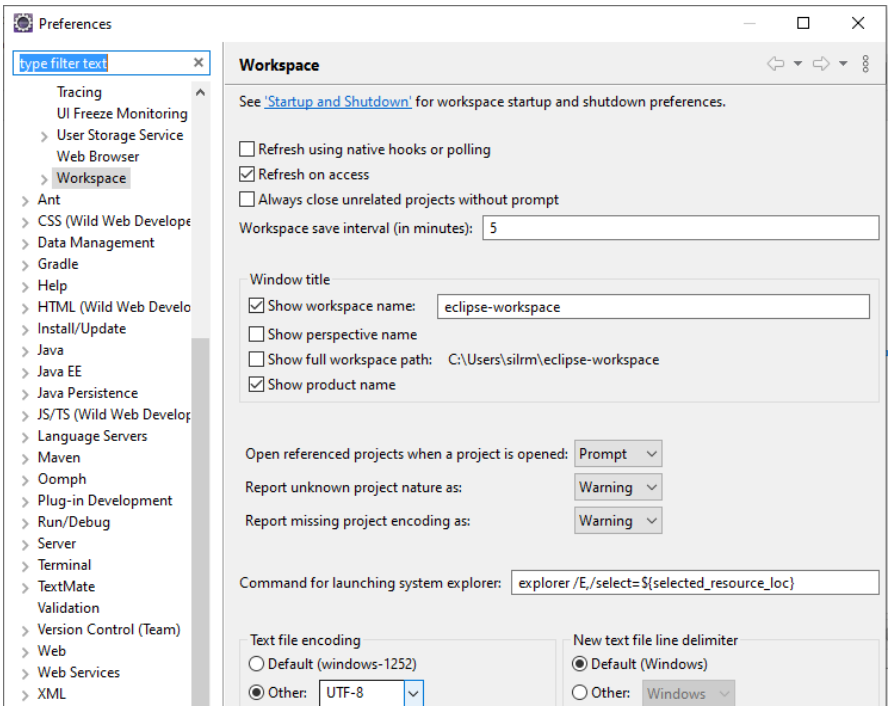
```
/**
* String: métodos úteis
*/
String disciplina = "Sistemas Orientados a Objetos I";
System.out.println("disciplina: " + disciplina);
// Isolando um caráter:
System.out.print("primeiro caráter: ");
System.out.println(disciplina.charAt(0));
System.out.print("segundo caráter: ");
System.out.println(disciplina. charAt(1));
// O primeiro caráter de uma String tem o
// índice 0, o segundo o índice 1 e assim por diante
// letra = 's';
char letra = disciplina. charAt(2);
// substrings:
System.out.print("primeiras cinco letras: ");
System.out.println(disciplina. substring(0, 5));
System.out.print("letras a partir da quarta: ");
System.out.println(disciplina. substring(4));

// número de carateres numa String:
System.out.print("tamanho da frase: ");
System.out.println(disciplina.length() + " letras");
// usando os carateres de tabulação e quebra
// de linha:
System.out.println(""
+ disciplina.length()
+ " letras"
+ " \n"
+ " Nova linha\ttabulação");
```

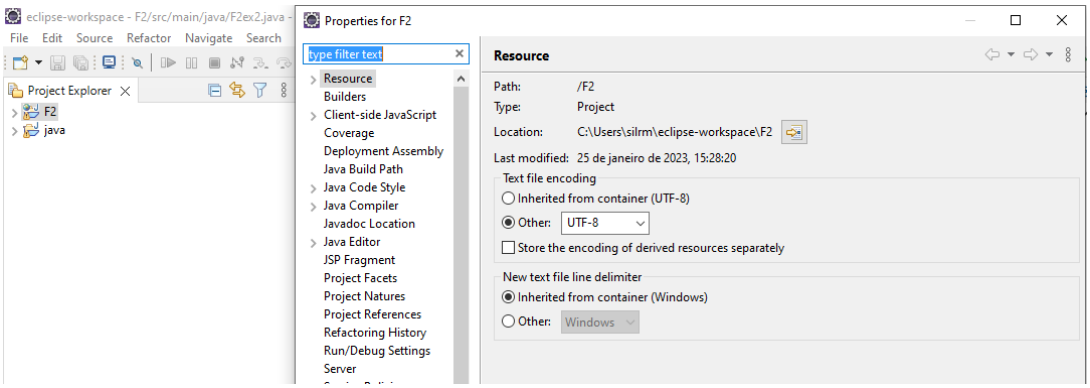
7. Escreve o código acima e observa o resultado.

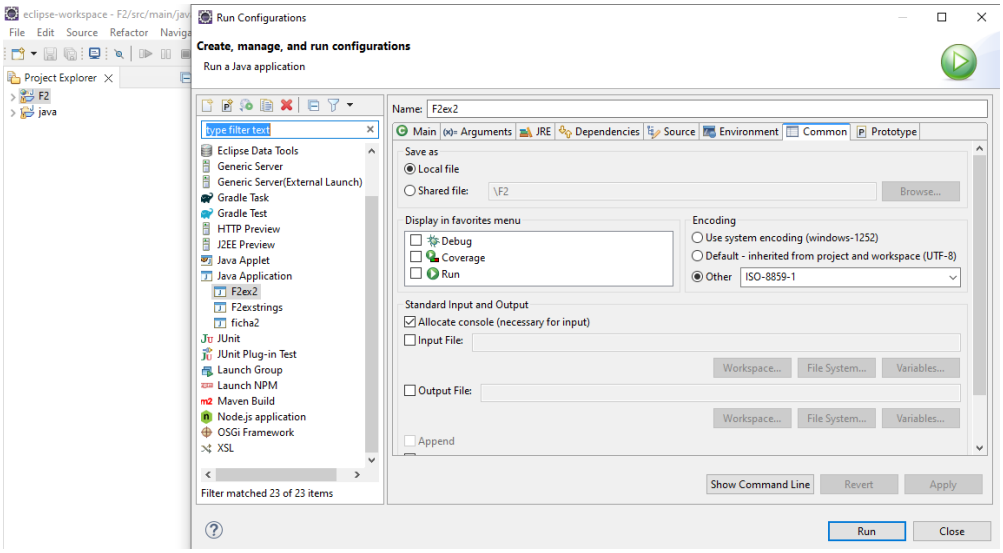
Configurar o encoding do Eclipse para UTF-8:

Menu Window->Preference ->workspace



Nas propriedades do projeto:





Colocar o encoding do projeto no formato ISO-8859-1

run -> run configurations -> java application -> nome do projeto -> common.