- Faça login no computador seguindo as instruções do docente.
- No directório *Desktop* vai encontrar um conjunto de ficheiros úteis para o exame.
- Utilize o executável `./run-jar` como complemento na especificação do programa.
  Exemplo: `./run-jar p1.txt`
- Pode consultar a documentação das classes `Java` usando o comando `view-javadoc`.
  Exemplo: `view-javadoc ParseTreeProperty`
- Tem à sua disposição os comandos de apoio à programação em `ANTLR4`:
  `antlr4`, `antlr4-build`, `antlr4-clean`, `antlr4-main`, `antlr4-test`
- Utilize o enunciado como rescunho, e no final **entregue-o com o cabeçalho preenchido.**
- Caso pretenda desistir deve indicar essa decisão no enunciado e executar o comando: `desisto`

Problema: Pretende-se implementar uma linguagem para uma calculadora que aceite números complexos. Como exemplo inicial, considere o seguinte programa:

```
## p1.txt
output 2+3i;    ## escreve na consola o número 2 + 3i
output 4;       ## escreve na consola o número 4
output i;       ## escreve na consola o número i
4-i => c;       ## guarda o número 4 − i na variável c
output c;       ## escreve na consola o número armazenado na variável c
```

Nota 1: Tente tirar o melhor proveito possível das instruções exemplificadas por forma a tornar a linguagem o mais genérica possível. No entanto, pode considerar que os números complexos literais (ex: 1, $-i$, $2+4i$) são sempre compostos por partes reais e imaginárias inteiras.

Nota 2: As variáveis são compostas por uma letra inicial, e depois uma qualquer sequência de letras ou dígitos.

Nota 3: Não se esqueça das verificações semânticas. Existem ficheiros `err?.txt` para o ajudar nesse fim.

a) Implemente em `ANTLR4`, uma gramática `CalComplex` para esta linguagem. [5 valores]

b) Implemente um interpretador que execute as instruções desta linguagem. [5 valores]

c) Altere a gramática e o interpretador por forma a permitir a realização das operações aritméticas básicas com complexos (com a precedência natural): soma ($+$), subtracção ($-$), multiplicação ($*$), divisão ($:$) e parêntesis (ver programa `p2.txt`)[1]. [4 valores]

```
## p2.txt
output i-4+5i+8;
5-2i => c1;
output (-3-i):c1;
4  + i-4*5i -(3i+5:4i) => c2;
output c2;
output c1*c2:c2+(i)-4;
```

d) Faça com que o interpretador leia o programa a partir de um ficheiro (cujo nome é passado como argumento do programa), e altere a gramática por forma a permitir a entrada de números complexos pelo utilizador. Esta entrada deve pedir separadamente a parte real e a parte imaginária do número (ver programa `p3.txt`). [2 valores]

[1] $(a+bi) \pm (c+di) = (a \pm c) + (b \pm d)i \qquad (a+bi) * (c+di) = (ac-bd) + (ad+bc)i \qquad \frac{a+bi}{c+di} = \frac{ac+bd}{c^2+d^2} + \frac{bc-ad}{c^2+d^2}i$

```
## p3.txt
output input "n";  ## (input) pede as partes reais e imaginárias de um número complexo
                   ## ao utilizador apresentando os textos "n(re): "e "n(im): ",
                   ## e (output) escreve o número na consola
(input "n1") :` ( input "n2" ) => div;
output div;
```

e) Acrescente as seguintes operações:

(a) Conjugado de números complexos `conj(number)` (ver programa `p4.txt`). [2 valores]

(b) Módulo de números complexos `|number|` (ver programa `p5.txt`). [2 valores]

18 de Junho de 2018