Universidade Federal de Campina Grande – UFCG
Centro de Engenharia Elétrica e Informática – CEEI
Unidade Acadêmica de Sistemas e Computação – UASC

Disciplina: Laboratório de Programação 2

Laboratório 01

Como usar esse guia:

- Leia atentamente cada etapa
- Dedique especial atenção à seção “Acompanhe seu aprendizado neste lab” antes, durante e depois de realizar esta atividade
- Quadros com dicas tem leitura opcional, use-os conforme achar necessário
- Preste atenção nos trechos marcados como importante (ou com uma exclamação)

Sumário

Acompanhe seu aprendizado neste lab

Conteúdo sendo exercitado

- Tradução de um programa Java (compilação + interpretação)
- Uso de pacotes e classpath
- Introdução à sintaxe de Java: desvio condicional, laços, operadores, comando de atribuição
- Introdução aos tipos primitivos
- Conceitos básicos de entrada/saída: uso da entrada (teclado) e saída (monitor) padrão

Objetivos de aprendizagem

- Resolver problemas de programação usando a sintaxe básica de Java. Isso inclui escrever o programa (criar uma classe, escrever o método main, usar estruturas básicas da linguagem, usar a entrada e saída padrão) e executá-lo (compilar + interpretar) usando o kit de desenvolvimento java (jdk).

Perguntas que você deveria saber responder após este lab

- Como é o processo de execução de um programa em Java?
- Dado um programa Java para escrever uma mensagem na tela, como o Hello.java, quais os passos para executar esse programa?
- Uma das vantagens de programar em Java é a portabilidade. Como é possível ver essa vantagem com os programas que você implementou neste lab?
- Quais as diferenças entre uma linguagem estaticamente e dinamicamente tipada? Como Java se define nesse contexto?
- Você consegue explicar detalhadamente a semântica do seguinte comando: System.out.println(“Hello”);
- Por que o método main é definido como static?
- Por que em Java costumamos organizar os programas em pacotes?

Material para consulta

- Referências bibliográficas para iniciar:
    - material de referência desenvolvido por professores de p2/lp2 em semestres anteriores
    - o livro Use a cabeça, Java
    - o livro Java para Iniciantes
    - os livros:
        - Core Java
            - https://plataforma.bvirtual.com.br/Acervo/Publicacao/1238
        - Java, Como programar (Deitel)
            - https://plataforma.bvirtual.com.br/Acervo/Publicacao/39590
- Praticar é essencial. Já instalou o ambiente de desenvolvimento Java (JDK java)?
- Este artigo explica como ocorre o processo de tradução de um programa em Java
    - Você sabia que Java agora tem um interpretador nativo (http://openjdk.java.net/jeps/222)?

    - Você sabia que temos um shell oficial para Java? O JShell! Digite `jshell` em um terminal para testar.
- Explorando melhor este conteúdo no livro Use a Cabeça Java:
    - Por que você deve organizar seu código em packages - Página 5 tópico Estrutura do código em Java".
    - Java é uma linguagem interpretada ou compilada? - Página 13 tópico "A máquina Virtual Java".
    - Como se define uma variável em Java? - Página 5 tópico "Aponte seu lápis".
    - Qual o comando em Java para imprimir algo na saída padrão? - Página 10 tópico "Aponte seu lápis"
    - Tipos de variáveis - Página 36 tópico "Declarando uma variável"
    - Por que usar Java. - Página 1 tópico "Introdução"
    - Java é uma linguagem estaticamente tipada? - Página 13 tópico "A máquina Virtual Java".
    - Por que usar static. - Página 6.

PARTE 1: Criando e executando programas Java

O primeiro código Java

Crie um diretório chamado exercicio e dentro dele um arquivo chamado IdadePreferencial.java com o seguinte conteúdo (você pode usar qualquer editor de texto de sua preferência):

```
public class IdadePreferencial {
    public static void main(String[] args) {  // Definindo uma função
        int idade = 20;
        if (idade >= 60) {
            System.out.println("Voce tem " + idade
                    + " anos. Voce pode usar o atendimento especial.");
        } else {
         System.out.println("Voce tem " + idade + " anos. Voce ainda nao
pode usar o atendimento especial.");
        }
    }
}
```

Este programa imprime na saída padrão uma frase indicando a sua idade e se você é uma pessoa que pode usar o atendimento preferencial (acima de 60 anos). Lembre-se de nomear seu arquivo corretamente, IdadePreferencial.java (*Iremos entender melhor a estrutura desse código mais adiante*).

Dicas - Novos termos

Em java, aprenderemos a programar de um jeito diferente… e programar de um jeito diferente exige novos termos e expressões. Abaixo algumas das palavras e ideias gerais dos novos

conceitos que iremos explorar. **Não se preocupe em entender tudo perfeitamente agora, nós iremos explorar melhor esses conceitos no futuro!** Por enquanto, apenas siga os próximos passos.	
public	Público → Significa que a entidade (função, classe, variável, etc.) é acessível por qualquer outra entidade.
class	Classe → É a base de organização de código Java.
static	Static → Significa que o código é acessível de imediato
// Definindo...	// é o marcador para comentários de uma linha no código Para comentários de múltiplas linhas usamos: /* TEXTO TEXTO TEXTO */
void	void → Define que a função não retorna valor. Em java você precisa definir o tipo de retorno das funções, dos argumentos, das variáveis. Veremos isso adiante. :)
String[]	String[] → Identifica o tipo array (lista de tamanho fixo) de palavras
int	int → O tipo inteiro
System.out	Entidade para trabalhar com a saída padrão
println	Método executado na entidade acima para imprimir uma linha
main	Método a ser executado quando o programa é iniciado.

Compilando código Java

Abra um terminal (ou console) no Linux, vá para o diretório onde o programa acima foi salvo (com o comando **`cd`**) e olhe o que tem dentro do diretório (com a comando **`ls`**). Apenas o código fonte (IdadePreferencial.java) estará lá.

Dicas - Comandos	
Para abrir um terminal	Comandos: gnome-terminal, kterm, xterm, Prompt (windows), ou Terminal (interface linux)

Para criar e acessar um diretório	Use o comando **`mkdir`** e para passar de um diretório para outro use o comando **`cd`**. Exemplo:
	`$ mkdir nomeDoDiretorio`
	`$ cd nomeDoDiretorio`
	`nomeDoDiretorio$ mkdir ...`

Para ver o conteúdo de um diretório	Comando **`ls`** (LS em minúsculo) ou **`dir`** (no windows)

Para mover um arquivo de um diretório para outro	Comando **mv** (no linux) ou **move** (no windows). exemplo: `$ mv arquivo nomeDoDiretorioDestino`

Agora tente compilá-lo usando o comando javac:

```
javac IdadePreferencial.java
```

O processo de **compilação** converte o arquivo de código-fonte para um código próximo ao da máquina que irá executar esse programa. Em java, existe uma máquina virtual (Java Virtual Machine - JVM) que executa código. Esse código da JVM chama-se bytecode.

O exemplo abaixo mostra parte do bytecode do programa IdadePreferencial, onde cada linha é um comando entendível e executado pela JVM. A JVM **interpreta** cada linha abaixo, realizando a operação necessária na memória, cpu ou nos dispositivos adequados.

```
0: bipush       10
2: istore_1
3: iload_1
4: bipush       60
6: if_icmplt    42
9: getstatic    #2
12: new         #3
15: dup
16: invokespecial #4
19: ldc         #5
...
```

Se seu programa não tiver erros de sintaxe, a compilação será bem sucedida e o arquivo de bytecodes (IdadePreferencial.class) vai ser gerado. Use novamente o comando **ls** para se certificar que o arquivo IdadePreferencial.class existe.

IdadePreferencial.class é o arquivo que contém os bytecodes do programa IdadePreferencial.java. Agora, a Java Virtual Machine (JVM) pode executar o seu programa interpretando blocos de comandos para a linguagem de máquina. Tente executar o seu programa, utilizando o comando:

```
java IdadePreferencial
```

Experimente mudar o nome da classe colocando todas as letras em minúsculo, sem modificar o nome do respectivo arquivo. O que acontece?
→ O erro indica que Java é case-sensitive (isso significa que Java diferencia letras maiúsculas de minúsculas).

Dicas - Erros frequentes neste exercício -

1. Nome do Arquivo
`Idadepreferencial.java:1: error: class IdadePreferencial is public, should be declared in a file named IdadePreferencial.java` `public class IdadePreferencial {`
Fique atento ao nome do arquivo! É importante que o nome seja igual ao nome da classe que está sendo declarada (no erro apresentado, o **p** está em minúsculo).
2. Compilador não instalado
`-bash: javac: comando não encontrado`
No caso de receber uma mensagem de erro como esta ou parecida, é provável que o compilador java não esteja instalado. O compilador java (javac) vem junto com o Java JDK (não confundir com o Java JRE).
3. Erro no código
`IdadePreferencial.java:10: error: reached end of file while parsing` `}`
Você provavelmente esqueceu de copiar uma chave no final.

Pacotes e Classpath

Vamos agora modificar o nome do programa para IdadePreferencialInteligente. Essa modificação requer que você (1) modifique o nome do seu arquivo para IdadePreferencialInteligente.java, caso contrário o compilador java não vai reconhecer o seu arquivo como um programa Java e não vai compilar seu código e (2) edite o código e modifique o nome da classe do programa.

```
mv IdadePreferencial.java IdadePreferencialInteligente.java
```

Dessa vez vamos começar a usar o conceito de pacote (package). Pacotes servem para organizar o seu código, assim como os diretórios servem para permitir organizar seus arquivos no seu computador. Para usar pacotes, a primeira linha do seu programa deve indicar o pacote a que o programa pertence, neste caso insira a seguinte linha no início do seu programa .java:

```
package lp2.lab01;
```

Para simplificar o uso de pacotes, crie a árvore de diretório lp2/lab01 a partir do diretório corrente. Use o comando **`mkdir`**, se estiver em um terminal Linux. Salve o programa IdadePreferencialInteligente.java no diretório lp2/lab01. Vá para o diretório lp2/lab01 e compile novamente o programa.

Vale lembrar que agora o bytecode gerado terá um nome completo, que inicia com o nome do pacote e estará numa árvore de diretórios idêntica ao que foi definido no pacote

(lp2/lab01/IdadePreferencialInteligente.class). Você terá que compilar novamente o seu código para que essa mudança aconteça (gerar um novo bytecode).

Você pode compilar sem ir ao diretório lp2/lab01, indicando apenas o caminho (*considere que agora seu diretório corrente é exercício*):

```
javac lp2/lab01/IdadePreferencialInteligente.java
```

Vamos executar indicando o caminho da classe pelo seu pacote. Tente executar:

```
java lp2/lab01/IdadePreferencialInteligente
```

Em alguns sistemas pode não ser possível encontrar a classe IdadePreferencialInteligente. Para algumas JVMs encontrarem a classe navegando pelos diretórios pode ser necessário usar **a notação de pacotes**. Caso não consiga executar seu programa com a linha anterior, tente executar:

```
java lp2.lab01.IdadePreferencialInteligente
```

Pode ser necessário ensinar à Java Virtual Machine (JVM) em que diretórios ela deve procurar os seus programas (esse diretório é o **classpath** ou **cp**). Por exemplo, saia dos diretórios do seu projeto java até o diretório pai de 'exercicio'. Agora tente executar sua classe usando apenas:

```
java lp2.lab01.IdadePreferencialInteligente
```

ou então:

```
java exercicio.lp2.lab01.IdadePreferencialInteligente
```

Note que a JVM não consegue encontrar a classe pois não sabe qual o diretório do projeto. Vamos usar o classpath para isso. Tente executar, ainda no diretório atual o seguinte comando:

```
java -cp ./exercicio lp2.lab01.IdadePreferencialInteligente
```

Agora sim! A JVM sabe que existe uma estrutura de pacotes Java no caminho "./exercicio".

PARTE 2: Programando em Java

Definição de Variáveis

Vamos modificar o programa IdadePreferencialInteligente.java para que ele calcule sua idade a partir do ano atual e do ano de nascimento. Nesse caso, você vai precisar criar uma variável para guardar

o valor da idade que será resultado da diferença entre dois literais inteiros, um representando o ano atual e, o outro, o ano de nascimento.

Java é uma linguagem **estaticamente tipada**, assim, todas as variáveis precisam ter associado um tipo em tempo de compilação. Esta associação é feita da seguinte forma:

```
<tipo_da_variável> <nome_da_variável>
```

Ex: `int x;`

No caso da idade o tipo a ser usado será inteiro (int), mas Java suporta vários outros tipos primitivos como mostrado no material disponível (ONLINE, LIVRO-UseCabecaJava, cap1, Livro-JavaIniciantes).

Para calcular a idade será necessário usar o operador de subtração de Java (-). A linguagem também dispõe de outros operadores aritméticos, tais como: soma (+), multiplicação (*) e divisão (/). Ver mais informações (ONLINE, LIVRO-UseCabecaJava, cap1, Livro-JavaIniciantes).

Agora para mostrar o valor da variável que representa a idade será necessário usar o operador de concatenação de Java (+). Note que é o mesmo operador da soma aritmética, assim, este operador é dito “sobrecarregado”, pois possui dois casos de uso. Então, a concatenação ocorre da seguinte forma:

```
int x = 10;
System.out.println("Número: " + x);
```

Nesse caso, se um dos operandos for textual, o operador + fará concatenação; se os dois operandos forem numérico, o operador + fará soma aritmética.

Faça as modificações no programa para que ele calcule idade a partir do ano atual e do ano de nascimento, compile-o e execute-o novamente.
OBS: Sem entrada de dados.

Comando de Desvio

No programa IdadePreferencialInteligente.java verificamos se a pessoa está apta a usar serviço preferencial. Para tal, usaremos as estruturas de desvio de Java.

Um comando de desvio define uma condição em um programa, que permite que grupos de comandos sejam executados de maneira condicional, de acordo com o resultado da avaliação de um determinado teste (verdadeiro ou falso). Ou seja, programas utilizam comandos de desvio para escolher entre cursos alternativos de ações. A sintaxe do comando de seleção em Java é:

```
if (expressão condicional){
   comando1;
}else{
```

```
    comando2;
}
```

O comando1, será executado se o resultado da avaliação de expressão for verdadeiro (digamos x > 3). Caso contrário (else) o comando2 será executado.

Importante: i) vários comandos podem ser executados na cláusula if ou na cláusula else, bastando para isso que sejam usados os delimitadores de escopo { } (abre chaves e fecha chaves). ii) a indentação não funciona em Java como delimitador de blocos de código (ou de escopo). Use chaves!

A expressão condicional é uma expressão booleana, podendo conter operadores relacionais (ex. >, maior; < menor) ou lógicos (ex. &&, and; ||, or). Mais operadores (ONLINE, LIVRO-UseCabecaJava, cap1, Livro-JavaIniciantes).

Observe que a mensagem apresentada pelo programa IdadePreferencialInteligente.java não contempla todos os casos de atendimento preferencial com exatidão, pois mulheres grávidas ou pessoas com pelo menos uma criança de colo, também tem preferência. Modifique seu programa para tratar esses casos, imprimindo qual o caso preferencial específico:
OBS: Sem entrada de dados.

```
Acima de 60 anos: "Preferencial Idoso."
Grávida: "Preferencial Gestante."
Criança de colo: "Preferencial pois está com x crianças de colo."
```
(onde x é a quantidade de crianças)

Além de um novo tipo de variável (boolean), você vai precisar usar comandos de desvio aninhados. Comandos de desvios “dentro” de outros comandos de desvio. Em Java, podemos utilizar diferentes maneiras para escrever comandos aninhados. Abaixo, apresentamos a maneira mais tradicional e que permite um melhor entendimento do código com comandos de desvio aninhamos. Sempre opte por menos linhas de código ;).

```
if(condicao1){
    comando1;
} else{
    if(condicao2){
        comando2;
    } else {
        comando3;
    }
}
```

OU

```
if(condicao1){
    comando1;
} else if(condicao2){
    comando2;
} else {
    comando3;
}
```

DIRLIDIDI. Primeiro Exercício do Lab 1

Importante!

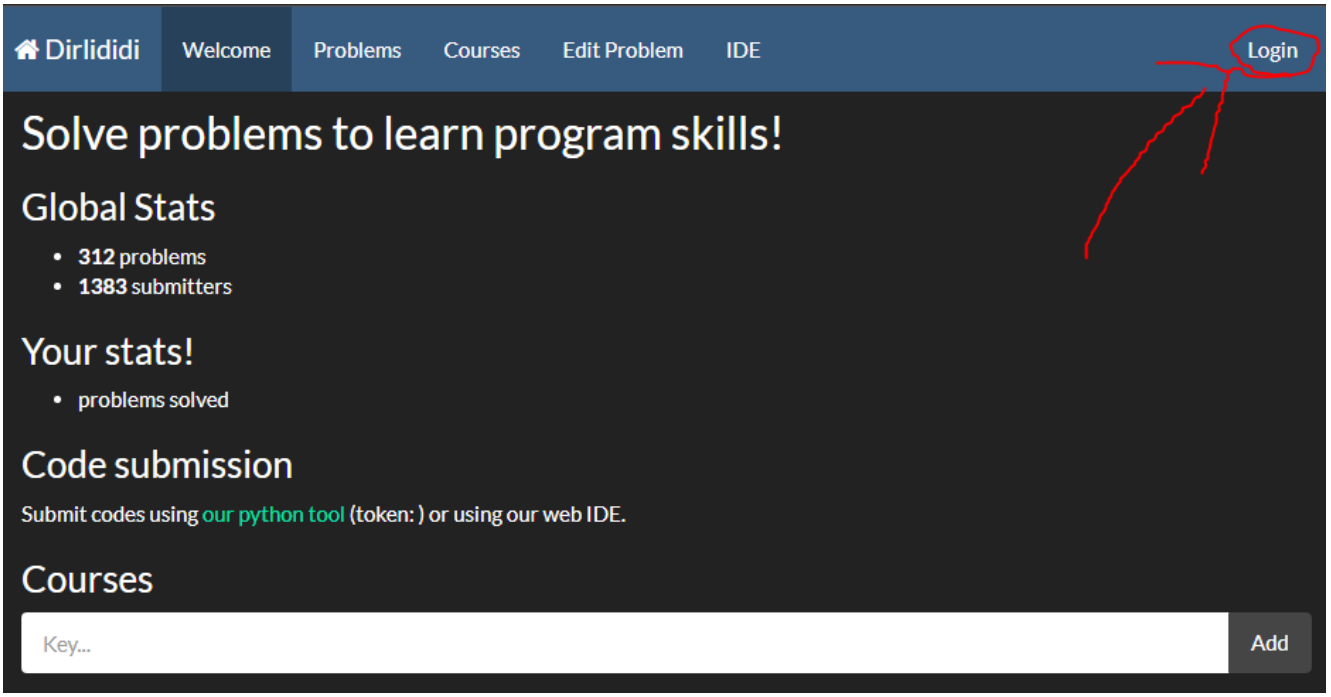
- No começo da definição de cada classe escreva o seguinte comentário abaixo:
- ```
  /**
   * Laboratório de Programação 2 - Lab 1
   *
   * @author Seu Nome Aqui - MATRICULA
   */
  ```

O Dirlididi (http://dirlididi.com) é um sistema online de submissão de exercícios. Esta será a plataforma utilizada para fazer o primeiro laboratório de programação.

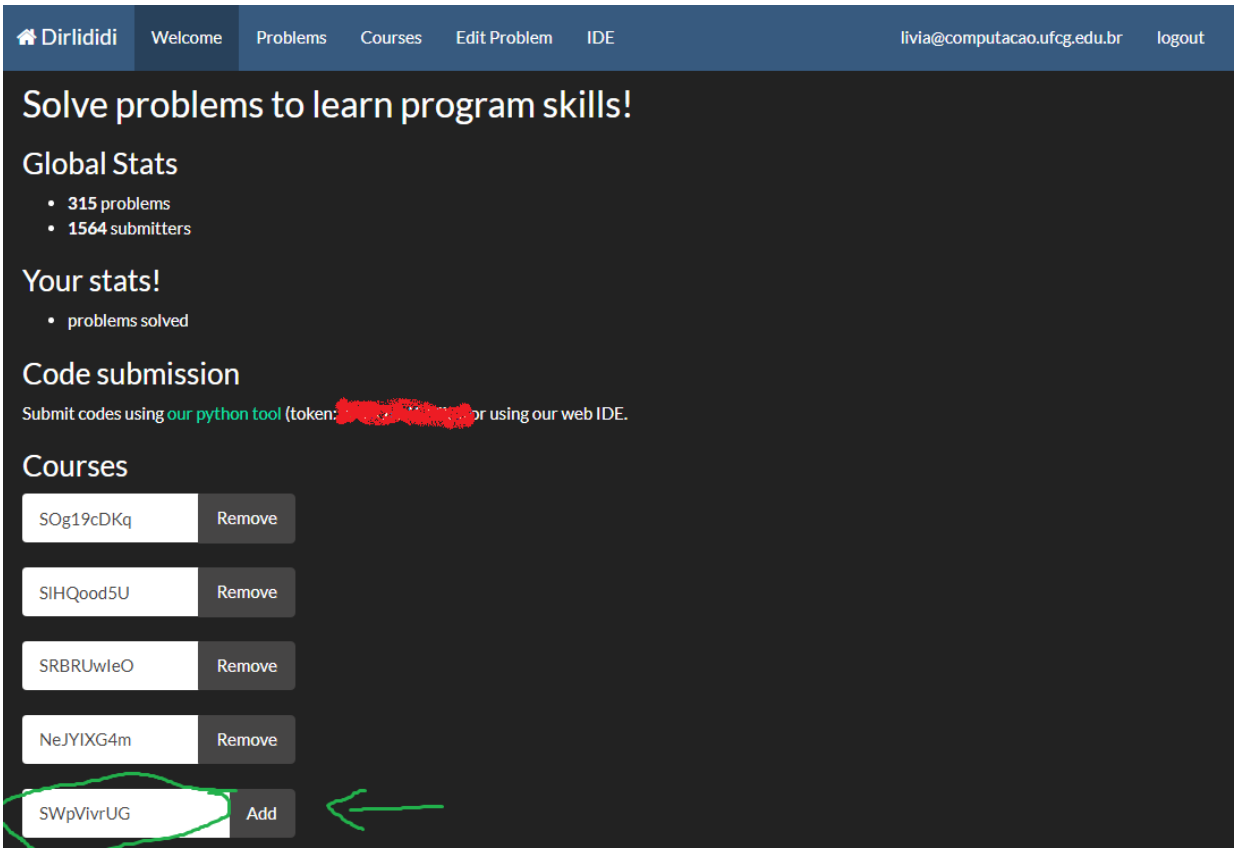
→ IMPORTANTE: LEMBRE DE SE ADICIONAR AO CURSO (ETAPA 2 ABAIXO) ←

Para isso, siga as etapas abaixo:

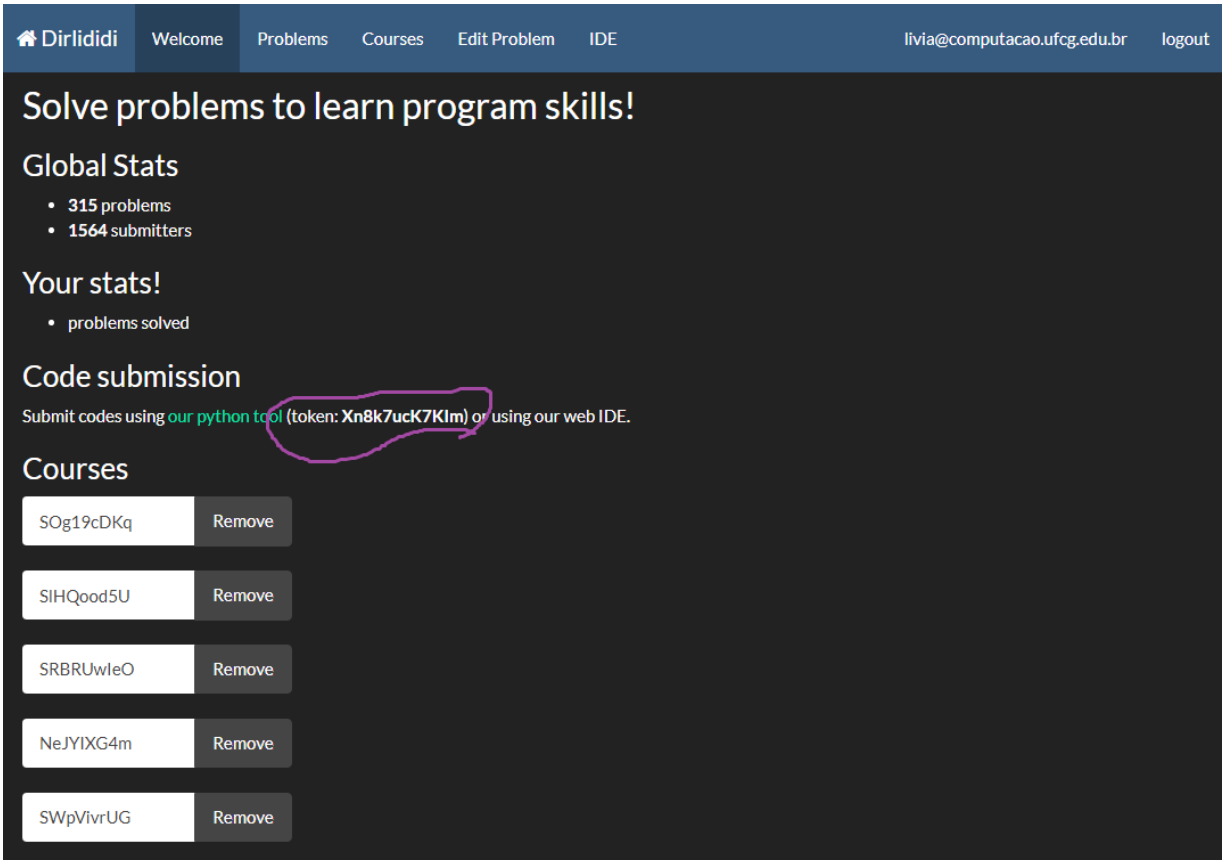
1. Faça login (canto superior direito) no site com sua conta CCC



2. Na página Welcome, adicione o curso (basta colocar a chave e clicar em Add) SWpVivrUG
  ```



3. Na página de entrada **anote** o TOKEN de submissão. Procure por ele na frase:
   *Submit codes using our python tool (token: **meutoken**) or using our web IDE.*



4. Baixe o script pyhton dirlididi.py e salve no diretório acima de seus programas Java.
   a. http://dirlididi.com/tools/dirlididi.py

b. Para submeter seu programa você usará o comando:
    i. python dirlididi.py submit **idquestao meutoken programa.java**
    ii. Você precisará do Python instalado em sua máquina (...já estão nos LCCs.)
5. Crie um programa Java sem pacote que imprima “Hello Dirlididi!” (sem aspas)
6. Submeta o programa:
    a. python dirlididi.py submit Q8lIaDijI **meutoken** Programa.java
        i. Q8lIaDijI é a identificação desse problema inicial;
            1. DICA: COPIE E COLE A IDENTIFICAÇÃO! ELA POSSUI L MINÚSCULOS E A VOGAL I MAIÚSCULA E ISSO GERA ERROS AO COPIAR O CÓDIGO!
        ii. **meutoken** é o token de submissão do seu usuário no dirlididi;
        iii. Programa.java é o arquivo com seu programa java.

Se o programa for executado com sucesso, você receberá a saída:

```
Results: .
```

Em caso de falha, uma mensagem irá dizer o motivo da falha. Você pode procurar por mais exercícios para fazer no próprio site do dirlididi (http://dirlididi.com/client/index.html#/problems)

Entrada Padrão

Em Java, o *System.out* é uma entidade que representa a saída padrão do programa. Nessa entidade é possível fazer a chamada ao método *println* que irá imprimir os argumentos passados a essa chamada em uma linha. Para ler a entrada padrão, é possível fazer uso do *System.in*. Essa segunda entidade, no entanto, disponibiliza apenas o método read para fazer leitura (que retorna o byte lido).

Veja o exemplo de leitura no programa LendoUmByte.java abaixo.

```
public class LendoUmByte {
    public static void main(String[] args) throws Exception {
        System.out.println("Digite um caractere:");
        System.out.println(System.in.read());
    }
}
```

Observe que o resultado aparece como o valor numérico do caractere lido. Isto não é algo muito amigável de se trabalhar. Para facilitar a leitura de dados, Java permite o uso de objetos próprios para isso, como o Scanner.

Tente executar o programa abaixo para ver um exemplo de uso do scanner para a leitura de dados.

```
import java.util.Scanner;

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

```
        System.out.println("Digite o seu nome:");
        String nome = sc.nextLine();
        System.out.println("Digite a sua idade:");
        int idade = sc.nextInt();
        System.out.println(nome + " - " + idade);
    }
}
```

É importante observar que para fazer uso de uma entidade externa, é necessário **importá-la** ao código. Esse processo é feito através do IMPORT. Java, por padrão, conta com algumas entidades auxiliares úteis como o System, Math, Integer, etc.

Em seguida, no começo do main, nós CRIAMOS o OBJETO **sc**. Um objeto é uma entidade que tem (encapsula) atributos e operações. Em Python, é muito comum fazer uso do objeto list (lista [ ]) que tem outros objetos/valores dentro dele e que tem operações como o append. Essas operações, quando são realizáveis nos objetos, são conhecidas como métodos. Um método é uma operação realizada em um objeto e que, em geral, altera o estado desse objeto.

É possível visualizar alguns exemplos de métodos sendo invocados no objeto **sc**, como o método **nextLine** (que pega uma linha) e o método **nextInt** (que pega um inteiro). Internamente o Scanner está fazendo várias operações de leitura no System.in para poder retornar o valor desejado ao usuário.

DIRLIDIDI. Segunda parte de exercícios do Lab 1

Usando o Dirlididi, faça as questões indicadas abaixo. Lembre de usar a instrução que segue para submeter seus programas:

python dirlididi.py submit **id_questão meutoken** Programa.java

QCT9g3lJ2 - Faça um programa que receba como entrada um número inteiro X e imprime como saída a mensagem “dobro: Y, triplo: Z” (sem as aspas). Por exemplo, para a entrada 2, a saída deve ser:

```
dobro: 4, triplo: 6
```

QN9tFkXbc - Faça um programa que recebe 2 números de ponto flutuante (dica: nextFloat do Scanner) e que imprime “pass: True!” se a média desses dois números for maior ou igual a 7.0 e imprime “pass: False!” caso contrário.

Entrada	**Saída**
8.0 9.0	pass: True!

Dica aqui: ao usar o método nextFloat seu programa vai tentar usar o formato do idioma em que seu sistema está configurado para reconhecer os números. Então pode ser que ele considere que um e

meio é 1,5 ou 1.5 dependendo dessa configuração. Isso porque em português é 1,5 e em inglês é 1.5 o formato. Mil pra a gente é 1.000,00 e em inglês é 1,000.00

Como o dirlididi está rodando em uma máquina configurada em inglês, ele espera 1.5. Se você está testando o programa na sua máquina e quer reproduzir esse comportamento, precisa passar uma opção para a JVM: em vez de por exemplo *java Media* use *java -Duser.language=en Media*

O74TDGIEa - Uma função monótona é uma função que tem o mesmo comportamento crescente ou decrescente de valores. Quando todos os valores da função aumentam e não são iguais entre si, chamamos essa função de estritamente crescente. Quando todos os valores da função diminuem (e também não são iguais entre si) chamamos essa função de estritamente decrescente.

É possível ter fortes indícios de uma função ser estritamente crescente ou decrescente observando alguns valores da função. Se todos os valores forem crescentes esta função pode vir a ser uma função monótona estritamente crescente. Se todos os valores forem decrescentes, esta função pode vir a ser uma função monótona estritamente decrescente.

Faça um programa que receba 4 valores e retorne as mensagens "POSSIVELMENTE ESTRITAMENTE CRESCENTE" e "POSSIVELMENTE ESTRITAMENTE DECRESCENTE" se os valores recebidos forem, respectivamente, todos crescentes e diferentes entre si, todos decrescentes e diferentes entre si. Caso nenhuma dessas condições seja satisfeita, imprima a mensagem: "FUNCAO NAO ESTRITAMENTE CRES/DECR". Veja os exemplos de entrada e saída abaixo.

Entrada	**Saída**
1 2 3 4	POSSIVELMENTE ESTRITAMENTE CRESCENTE
4 3 2 1	POSSIVELMENTE ESTRITAMENTE DECRESCENTE
4 4 3 2	FUNCAO NAO ESTRITAMENTE CRES/DECR

R7qL9aIKS - Construa um programa de calculadora onde o usuário irá escolher uma das 4 operações básicas abaixo como ação do usuário. Caso o usuário digite um dos símbolos: +-*\ o programa deve consumir duas entradas seguintes. Se o usuário digitar qualquer outra entrada, deve apenas exibir: ENTRADA INVALIDA e terminar o programa.

Após consumir as duas entradas, o programa deve exibir o resultado da operação com a mensagem: RESULTADO: X onde X é o resultado da operação sobre as duas entradas de valores. Isto será exibido em todas as situações, exceto quando o segundo valor de entrada for 0 e a operação for de divisão. Neste caso, é exibida a mensagem "ERRO".

Entrada	**Saída**

+ 2,0 3,5	RESULTADO: 5,5
/ 5,0 0,0	ERRO
0	ENTRADA INVALIDA

Dicas - Comparando Strings
Para comparar duas strings use o método equals. Exemplo: “banana”.equals(“banana”)
Se o seu sistema estiver em inglês você deve usar . (ponto) no lugar de vírgula como separador de ponto flutuante.

Comandos de Laço

Java tem diferentes maneiras de operar com repetição. Existem 2 comandos principais que exploraremos nessa disciplina. O primeiro é o for e o segundo é o while. Uma delas é o for-each, já conhecido dos programadores Python.

```
for (String palavra : linha.split(" ")) {  // o split quebra uma string em uma sequencia de strings
    System.out.println("Palavra: " + palavra);
}
```

A segunda maneira é definindo uma variável de interação e controlando seu incremento, como mostra o exemplo abaixo:

```
for (int i = 1; i < 10; i++) {
    System.out.println("Tabuada de 5: 5 * " + i + " = " + (5 * i));
}
```

Este segundo formato sempre tem o mesmo padrão:

```
for (DECLARAÇÃO DE VARIÁVEL; CONDIÇÃO DE PERMANÊNCIA; PASSO DE ITERAÇÃO);
```

Observe que “i++” é uma operação válida em Java e é equivalente a fazer “i=i+1”. Você pode omitir qualquer uma dessas etapas. Por exemplo, se não quiser declarar uma variável, mas fazer reuso de alguma existente, basta usar o for sem o primeiro elemento (ex: for (;i<10; i = i + 2)).

Quando não conhecemos a quantidade de passos de uma interação, é recomendado fazer uso do while que, semanticamente, é equivalente ao comando em Python. Veja o exemplo abaixo.

```
while (i > 10) {
    if (i % 2 == 0) {
        i = i / 2;
    } else {
        i = 3 * i + 1;
    }
}
```

Arrays

Java conta com diferentes estruturas de dados. Uma das estruturas mais básicas é o array. Um array é uma lista de elementos de tamanho fixo. É uma estrutura extremamente eficiente, pois todos esses elementos estão alocados na memória de forma sequencial. Entretanto, por causa dessa forma de alocação, o array precisa ter um tamanho fixo sempre.

No futuro aprenderemos outras estruturas de dados em java, como Listas, Conjuntos e Mapas (o equivalente a dicionários).

Arrays tem uma representação especial em java, e são definidos como apresentados abaixo:

```
String[] palavras = new String[10];  // Array de 10 strings

int[][] matriz = new int[4][2];   // Array multidimensional (matriz) de 4
linhas e 2 colunas

String[][] tabela = new String[10][];  // você pode definir o tamanho de
cada String[] interno no futuro

int[] exemplo = { 1, 2, 3, 4 };  // pode inicializar elementos já de cara
```

Atribuindo e acessando elementos:

```
int[] exemplo = { 1, 2, 3, 4 };
System.out.println(exemplo[0]);  // elemento de valor 1
exemplo[0] = 2;  // exemplo agora é {2, 2, 3, 4}
System.out.println(exemplo.length);  // o tamanho do array
```

DIRLIDIDI. Terceira parte de exercícios do Lab 1

PoFnkEzvk - Para avaliar uma população de dados é comum olhar o comportamento e características de elementos que estão acima (ou abaixo) da média. Faça um programa que receba, em uma linha, uma lista de inteiros e que imprima, na saída, apenas os valores que estão acima da média.

Seu programa deve imprimir os resultados em uma única linha e na mesma ordem que foram inseridos. Valores iguais a média não devem ser impressos e considere que para toda entrada, haverá pelo menos um valor na saída. Veja o exemplo.

Entrada	**Saída**
5 2 4 3 1	5 4

SIfR6pBmC - Wally mudou de nome. Felizmente, para a interpol, ele adotou um novo nome qualquer de 5 letras.

Para ajudar a desmascarar Wally, faça um programa que leia listas contendo nomes de pessoas. E imprima o último nome que pode ser o novo nome de Wally. Se não existir nenhum nome possível, imprima apenas o caractere ?. O programa termina quando encontrar uma linha com a palavra wally. Veja o exemplo.

Entrada	**Saída**
diego thiago arthur pedro abdias isaac matheus jorge joao nestor avelange ezequiel thor epaminondas wally	pedro jorge ?

NIGzVluci - Um professor gerou uma lista com os alunos e as notas das provas (0 a 1000) desses alunos. Ele deseja saber informações gerais sobre a prova e pede que você calcule: maior nota, menor nota, média, número de alunos com nota acima ou igual a 700, número de alunos com nota abaixo de 700. O programa para ao ler uma linha com um traço (-).

Veja o exemplo abaixo para entender como deve ficar a saída do seu programa. Imprima sempre a média truncada.

Entrada	Saída
pedro 1000 arthur 550 ana 920 jorge 700 -	maior: 1000 menor: 550 media: 792 acima: 3 abaixo: 1

Entrega

Resumindo,

1. Você deve fazer, em java, os problemas descritos abaixo no Dirlididi
2. Lembre de se cadastrar na turma "**SWpVivrUG**" (isso pode ser feito a qualquer momento antes da data de entrega -- teremos acesso aos códigos submetidos mesmo antes de você se cadastrar na turma).

Código	**Descrição**
Q8lIaDijI	Hello Dirlididi!
QCT9g3lJ2	Dobro e Triplo
QN9tFkXbc	Passou ou não.
O74TDGIEa	Função Monótona
R7qL9alKS	Calculadora
PoFnkEzvk	Acima da média
SlfR6pBmC	Wally!
NIGzVIuci	Alunos e notas

Sinta-se a vontade para fazer mais exercícios! No dirlididi tem mais problemas aqui. Alguns problemas legais e recomendados: Q36VouKXo, QIXAfeChM, SlfR6pBmC, PmZq7rg1I, MP51Jj5V2