FEUP	FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO Mestrado em Engenharia Informática e Computação Fundamentos da Programação	Auto Teste AT02 FACULTATIVO Duração: 60 min.

NOTAS IMPORTANTES:

1 - Deve respeitar rigorosamente os nomes dos procedimentos que são indicados bem como os formatos de saída dos resultados.

2 - Não use nunca carateres acentuados, nem nos nomes dos procedimentos nem dos parâmetros.

3 - Utilize comentários só "with Semicolons" e nunca "with a Box".

4 - O código desenvolvido durante a realização da prova, contido num único ficheiro com a extensão ".scm", deve ser submetido no Moodle usando o "link" correspondente à prova realizada. A não observação desta regra levará a que o código submetido não possa ser avaliado.

Cálculo da classificação

Da ficha de disciplina FP, foi extraído o seguinte:

Fórmula de cálculo da classificação final
Classificação = APP * 0.55 + AD * 0.05 + PE * 0.40
APP = médias das 3 melhores classificações obtidas nas provas práticas {PP!, PP2, PP3, PP4}

As várias componentes de avaliação, numa escala de 0 a 20:
- **AD** - Apreciação do desempenho dos alunos face aos exercícios propostos à turma;
- **PP1, PP2, PP3, PP4** - Realização de provas práticas em computador;
- **PE** - Realização de uma prova escrita com consulta.

Observações:
1- É condição de aprovação a obtenção de uma classificação mínima de 40% (8 em 20 valores) na componente PE.
2- ...

Condições para obtenção de frequência
Não exceder o limite de faltas ... e obter uma classificação mínima de 40% (8 em 20 valores) na componente APP.

1-
O procedimento **visu-classifica** tem um parâmetro, **cla**, e visualiza a classificação como se ilustra a seguir.

```
> (visu-classifica 10)
classificacao = 10        <-- observe UM espaço ANTES e DEPOIS do caráter =

> (visu-classifica 15.3)
classificacao = 15.3      <--- visu-classifica termina com newline

>
```

Complete o procedimento **visu-classifica**:

```
(define visu-classifica
 (lambda (cla)
     ...
```

2-
O procedimento **calcula-classifica** tem 6 parâmetros, **pp1**, **pp2**, **pp3**, **pp4**, **ad** e **pe**, com o significado indicado na ficha da disciplina, e devolve a classificação do aluno, calculada pela fórmula respetiva.

```
> (calcula-classifica 20 20 20 20 20 20)
20.0      <-- não é a visualização com display, mas o valor devolvido pelo procedimento
> (calcula-classifica 18 16 8 17 17 17)
17.0
> (calcula-classifica 10 10 10 10 10 0)
6.0
>
```

Complete o procedimento **calcula-classifica**:

```
(define calcula-classifica
 (lambda (pp1 pp2 pp3 pp4 ad pe)
     ...
```

(v.s.f.f.)

3-
O procedimento **classificacao** tem 6 parâmetros, **pp1**, **pp2**, **pp3**, **pp4**, **ad** e **pe**, com o significado indicado na ficha da disciplina, e deve:

1- verificar se tem a condição de frequência;
2- verificar se tem a condição mínima na **pe**;
3- calcular e visualizar a classificação, se se verificarem as duas condições anteriores.

Se não cumprir a condição de frequência, visualiza com display:
classificacao = sf **<--- "classificacao = sf"** ***seguido de newline***

Se cumprir a condição anterior,
mas não cumprir a condição mínima na **pe**, visualiza com display:
classificacao = pe **<--- "classificacao = pe"** ***seguido de newline***

Se cumprir as duas condições anteriores, visualiza com display
classificacao = cla **<--- "classificacao = "** ***seguido de cla e de newline,***
em que cla é o valor devolvido por calcula-classifica

Exemplos:

```
> (classificacao 0 7 8 8 8 15)
classificacao = sf

> (classificacao 8 8 8 8 8 7.9)
classificacao = pe

> (classificacao 18 16 8 17 17 17)
classificacao = 17.0

> (classificacao 8 8 8 8 8 8)
classificacao = 8.0
>
```

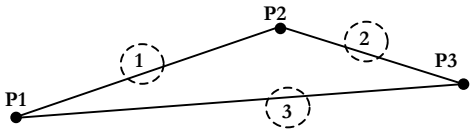
Complete o procedimento **classificacao**, podendo supor que lhe são fornecidos os procedimentos **visu-classifica** e **calcula-classifica**:

```
(define classificacao
 (lambda (pp1 pp2 pp3 ad pe)
     ...
```

Triângulos

O comprimento do segmento definido pelos pontos P1(x1, y1) e P2(x2, y2) pode ser determinado pelo procedimento **comprimento**, com os parâmetros **x1**, **y1**, **x2** e **y2**.

```
(define comprimento
 (lambda (x1 y1 x2 y2)
   (let ((x1-menos-x2 (- x1 x2))
         (y1-menos-y2 (- y1 y2)))
     (sqrt (+ (* x1-menos-x2 x1-menos-x2)
              (* y1-menos-y2 y1-menos-y2))))))
```



Considere agora um triângulo definido pelos vértices P1(x1, y1), P2(x2, y2) e P3(x3, y3).

5- O procedimento **lado-mais-comprido** tem como parâmetros **x1**, **y1**, **x2**, **y2**, **x3** e **y3** que representam as coordenadas dos vértices de um triângulo. Este procedimento devolve 1, 2 ou 3, conforme for o lado mais comprido do triângulo. Ver, na figura, como se associam 1, 2 e 3 aos lados do triângulo. Se não existir um lado mais comprido que todos os outros, o procedimento devolve 0.
Completar este procedimento, sabendo que usa o procedimento **comprimento**:

```
(define lado-mais-comprido
 (lambda (x1 y1 x2 y2 x3 y3)
     ...
> (lado-mais-comprido 0 0 2 2 4 0)
3
> (lado-mais-comprido 0 0 2 3 4 0)
3
> (lado-mais-comprido 0 0 1 2 2 0)
0
>
```

---------- FIM da Prova Prática ----------