PF' - EDL - 2018.2

Nome:

Tempo de prova: 90 minutos

Escolha 4 das 5 questões. Indique **na frente da prova** a questão não escolhida.

Cada outra questão (ou item) deixada em branco vale 1/4 da sua pontuação máxima. (Se for o caso, indique **na frente da prova** as questões (ou itens) deixadas em branco.)

---

1. (2.0 pontos)

Considere o trecho de código em C a seguir:

```
typedef union {
    Mago      m;  // contem o campo "mana"
    Guerreiro g;
    Arqueiro  a;
} Personagem;

Personagem get_personagem (void); // implementada em biblioteca externa

int main (void) {
    Personagem mago = get_personagem();
    printf("Mana: %d\n", mago.m.mana);
    return 0;
}
```

a. (0.5) O trecho de código acima compila? Por quê o conceito de `union` de C é considerado um exemplo de tipagem fraca?
b. (1.5) Qual é a técnica de programação usada para amenizar essa deficiência? Modifique o código acima usando essa técnica.

---

2. (2.0 pontos)

O conceito de *binding* amarra um nome a um atributo em um programa. A amarração pode ocorrer em *tempos* diversos, tais como *design da linguagem*, *implementação da linguagem*, *pré-processamento*, *compilação*, *ligação*, *carregamento*, ou *execução*.

Considere o programa a seguir:

```
#include <stdio.h>
#include <math.h>
#define PI 3.14
static int v = 10;
int f (void);
int main (void) {
    uint8_t x = sin(PI) + v + f();
    return x;
}
```

Indique o tempo em que cada propriedade é amarrada ao nome nos casos a seguir:

a. Valor de `PI`
b. Endereço de `v`
c. Tamanho de `int`
d. Implementação de `f`
e. Tipo de retorno de `f`
f. Tamanho de `uint8_t`
g. Endereço de `x`
h. Semântica de `+`
i. Implementação de `sin`
j. Tipo de `sin(PI)+v+f()`

Justifique a resposta para cada item.

---

3. (2.0 pontos)

Segundo Matthias Felleisen, o *poder de expressividade* de linguagens de programação é um conceito comparativo:

> A linguagem A é estritamente mais expressiva que a linguagem B se as seguintes condições forem verdadeiras:
>
> - Qualquer programa escrito em B pode ser reescrito em A mantendo a estrutura essencial do programa original.
> - Alguns programas em A não podem ser reescritos em B sem violar a estrutura essencial do programa original.

a. (0.5) O que você entende por *manter a estrutura essencial do programa*?
b. (1.5) Escolha duas linguagens e faça uma análise do poder de expressividade delas. **Use e analise exemplos de código.** (Recomenda-se analisar uma funcionalidade particular e não as linguagens inteiras.)

---

4. (3.0 pontos)

Considere as seguintes definições em Haskell para os tipos `Aluno` e `Turma` e também um exemplo de uma possível turma:

```
type Aluno = (String, Float, Float) -- Aluno é um tipo tupla com o nome e as duas notas
type Turma = [Aluno]                -- Turma é um tipo lista de alunos

-- Exemplo de turma:
turma1 :: Turma
turma1 = [ ("Joao",7,4), ("Maria",10,8), ("Jose",5,3), ... ]
```

Considere as assinaturas para as funções (já definidas) `map`, `filter`, e `foldr` a seguir:

```
map    :: (a->b) -> [a] -> [b]       -- mapeia lista de a's para lista de b's
filter :: (a->Bool) -> [a] -> [a]    -- filtra lista de a's para lista de a's
foldr  :: (a->b->b) -> b -> [a] -> b -- reduz lista de a's para valor de b
```

Para as questões a seguir, evite a definição de funções recursivas em favor das funções `map`, `filter`, e `foldr`. Além disso, tente reusar as funções já definidas sempre que possível.

Dicas:

- A função `length` calcula o tamanho de uma lista.
- A função `fromIntegral` converte um `Int` para um `Float`.

a. (0.5) Defina o **tipo** e a **implementação** da função `medias`, que recebe uma turma e retorna a lista de médias das provas de cada aluno. Exemplo de uso:

```
medias turma1   -- [5.5, 9.0, 4.0, ...]
```

b. (1.0) Defina o **tipo** e a **implementação** da função `resumo`, que recebe uma turma e retorna uma tripla com a média de todas as P1, média de todas as P2 e médias de todas as provas. Exemplo de uso:

```
resumo turma1   -- (7.33, 5.0, 6.16)
```

c. (1.5) Defina o **tipo** e a **implementação** da função `pretty`, que recebe uma turma e retorna uma string em que cada linha representa um aluno no formato `nome media (resultado)`. (Assuma a média 5.0 para aprovação.) Exemplo de uso:

```
pretty turma1

Joao 5.5 (aprovado)
Maria 9.0 (aprovado)
Jose 4.0 (reprovado)
```

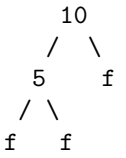
---

5. (3.0 pontos)

Considere a definição para árvore binária a seguir:

```
data Arvore = Folha | Galho Arvore Int Arvore
```

a. (0.5) Usando o tipo acima, defina uma árvore `a0` com a seguinte forma:



b. (1.0) Defina o **tipo** e a **implementação** da função `soma`, que recebe uma árvore e retorna a soma de todos os seus valores inteiros.

c. (1.5) Defina o **tipo** e a **implementação** da função `combina`, que recebe duas árvores e as combina em uma nova árvore que soma os valores inteiros em posições iguais. Exemplo: