Universidade Federal da Fronteira Sul

Ciência da Computação

Programação I - Prof. Fernando Bevilacqua

Trabalho 2 (T2) [Peso K=3]

Data de entrega: 29/05 (sexta-feira) e 22/06 (segunda-feira)

Descrição

Etapa 1 (entrega 29/05)

Nessa etapa o grupo de trabalho (dupla ou trio) deve ser o mesmo que implementou o interpretador no T1. Elaborar dois modelos UML referentes ao interpretador criado pelo seu grupo no trabalho 1 (T1). O primeiro diagrama UML deve descrever a modelagem atual do seu interpretador, ou seja, as classes e relações que foram usadas para a criação do interpretador do T1. Esse diagrama UML deve ser entregue em formato PNG, adicionado em uma pasta chamada `UML` no repositório do interpretador no Github.

O segundo diagrama UML deve conter uma nova modelagem para o interpretador. Essa seria a estrutura usada pelo seu grupo se vocês tivessem que criar um interpretador novo, a partir do zero, hoje. Aprenda com os erros e acertos realizados durante o T1 para modelar um interpretador melhor e mais fácil de ser implementado.

Etapa 2 (entrega 22/06)

Nessa etapa o grupo de trabalho pode ser uma dupla ou trio, (opcionalmente) diferente daquela que Implementou o T1. Durante essa etapa, a equipe fará melhorias no interpretador de outro grupo. O seu grupo receberá o nome e o link para o repositório do interpretador que será melhorado no dia 29/05, sexta-feira.

Seu grupo deverá fazer o seguinte:

- Fazer um ***fork*** do repositório do interpretador alvo;
- Implementar **todas** as melhorias obrigatórias existentes na *Tabela de novas funcionalidades*;
- Apresentar em aula o novo interpretador, descrevendo como as melhorias foram implementadas.

Avaliação

Os seguintes elementos serão considerados durante a correção:

- Funcionamento correto (o programa precisa cumprir seu objetivo conforme a descrição do trabalho);
- Legibilidade do código (nomes de classes com a primeira letra maiúscula, métodos e propriedades no formado nomeFormaCamelo, **indentação correta**, etc). Trabalhos mal indentados sofrerão desconto considerável em sua nota;
- Utilização de forma satisfatória dos conceitos de orientação a objetos vistos em aula;
- Comentários (o código fonte deve conter um bloco de comentário no começo informando o propósito do programa e o nome/email do seu autor).
- Histórico do controle de versão é condizente com o trabalho realizado (Ex. vários commits espalhados ao longo do tempo, não um único commit com diversos arquivos).

Observações

- **Não será possível entregar qualquer etapa do trabalho fora do prazo. O não cumprimento do prazo de qualquer uma das etapas implica em nota zero para todo o trabalho 2 (T2);**
- Trabalhos copiados receberão nota zero e o nome dos envolvidos será levado ao colegiado do curso;
- Programas que não compilarem receberão nota zero instantânea (nenhuma avaliação será realizada).

Universidade Federal da Fronteira Sul

Ciência da Computação

Programação I - Prof. Fernando Bevilacqua

Abaixo encontra-se uma tabela com os melhoramentos que devem ser implementados. O grupo precisa implementar todas as funcionalidades que forem obrigatórias.

Tabela de novas funcionalidades

Funcionalidade	**Implementalçao obrigatória?**

Expressões lógicas e aritméticas com N operandos e precedência, incluindo parênteses, usáveis em atribuições, laços e controladores de fluxo. Ex.:
 	
`a = b * (3 + 9) * 8;`
 	
`if(a + 3 < b * 4) {`
 	
`}`
 	
`while(x < a + (b - 2)) {`
 	
`}`
	Sim
	Se esse item não for implementado, o desconto de nota é de **-3.0**

Vetores, incluindo acesso a elementos com expressões aritméticas. Ex:
 	
`v[3] = 20;`
`v[i + 3*2] = 10;`
	Sim
	Se esse item não for implementado, o desconto de nota é de **-2.0**

Funções (sem parâmetro e sem retorno)	**Não** Se implementado, esse item rende **+1.0** na nota
Funções (com parâmetro ou retorno)	**Sim** Se esse item não for implementado, o desconto de nota é de **-3.0**

Structs no estilo:
 	
`var a = {campo1, campo2};`
`a.campo1 = 20;`
`a.campo2= 9;`
 	
Se o programador tentar acessar um tempo da struct que não existe, o interpretador deve informar um erro. Usando o exemplo acima, a linha abaixo causaria um erro:
 	
`a.campo99 = 9; // campo 'campo99' não existe.`
	Sim
	Se esse item não for implementado, o desconto de nota é de **-2.0**

Forma de entrega

O trabalho deve ser desenvolvido em um sistema de controle de versão online, como Github. O repositório deve ser um *fork* do projeto que está sendo melhorado.