1 Instruções Importantes

Nessa seção são apresentadas diversas informações relevantes referentes a entrega do trabalho e orientações a serem seguidas durante a implementação do mesmo. Leia atentamente antes de começá-lo.

1.1 Equipe de Desenvolvimento

O trabalho será desenvolvido individualmente ou em dupla.

1.2 Escolha da Linguagem de Programação

O trabalho deverá ser desenvolvido na linguagem Java.

1.3 Artefatos a Serem Entregues

Os artefatos a serem entregues são:

- código fonte do programa;
- arquivo de *build* para geração automática dos analisadores e compilação do fonte (makefile, ant, shell script, etc);
- documentação do trabalho em formato pdf.

Antes de enviar seu trabalho para avaliação, assegure-se que:

1. seu código compila e executa em ambiente Unix / Linux. Programas que não compilam receberão nota zero;
2. todos os fontes a serem enviados têm, em comentário no início do arquivo, nome e matrícula do autor do trabalho;
3. arquivo de documentação tenha a identificação do autor do trabalho;
4. arquivo compactado com os artefatos estão devidamente identificados com nome e matrícula.

1.4 Critérios de Avaliação

A avaliação será feita mediante análise do código fonte, documentação e apresentação do trabalho (entrevista). Os seguintes fatores serão observados na avaliação do código fonte: corretude do programa, estrutura do código, redigibilidade e legibilidade. A corretude se refere à implementação correta de todas as funcionalidades especificadas, i.e., se o programa desenvolvido está funcionando corretamente e não apresenta erros. Os demais fatores avaliados no código fonte são referentes a organização e escrita do trabalho.

A documentação do código deve conter informações relevantes para compilar, executar e auxiliar no entendimento do código fonte. Ressalta-se que na documentação não deve conter cópias do fonte – afinal o seu fonte é um dos artefatos enviado, mas deve apresentar as decisões de projetos tomadas: estruturas de dados usadas para análise e verificação de tipos, estratégia usada para verificar a semântica estática da linguagem (*static semantics*), dentre outras informações.

O trabalho deverá ser apresentado ao professor da disciplina e, só será avaliado após a realização da entrevista, i.e., trabalhos que não forem apresentados não terão nota. Na entrevista, o discente deverá elucidar, ao menos, como modelou e resolveu o problema, os resultados e conclusões obtidas. A entrevista

também tem a finalidade de avaliar a confiabilidade e segurança do autor do código em explicar pontos relevantes do trabalho desenvolvido.

Assim, a entrevista influenciará na avaliação dos artefatos entregues. Portanto, a nota final será dada a partir da avaliação do conjunto do código fonte, documentação e entrevista. **É de responsabilidade do discente solicitar a marcação do dia e horário da entrevista com o professor da disciplina**.

Atrasos serão penalizados por uma função exponencial de dias de atrasos, i.e., será reduzido do percentual da nota a exponencial na base 2 dos dias de atraso. A tabela a seguir mostra a nota em função dos dias de atraso:

Dias de Atraso	Nota
1	n*0.98
2	n*0.96
3	n*0.92
4	n*0.84
5	n*0.68
6	n*0.36
7	0

Observe que a partir do 7º dia de atraso seu trabalho não será mais avaliado.

2 Especificação Técnica do Trabalho

Na sequência para construção do compilador da linguagem *lang*, a próxima etapa é checar se os programas aceitos pelo analisador sintático usam operações sobre os tipos da linguagem corretamente. Portanto, nessa etapa pede-se que seja implementado uma verificação na árvore de sintaxe abstrata (AST) produzida pelo analisador sintático. Para isso, o analisador sintático produzido na etapa anterior deverá produzir como saída uma AST do programa de entrada. A AST será analisada a fim de coletar informações sobre os tipos das expressões e, a partir dessas informações, verificar-se-á o correto uso das construções da linguagem conforme seu sistemas de tipos.

As construções sintáticas de *lang*obedecem as regras usuais de linguagens de programação, com algumas pequenas exceções. A seguir, apresentamos, de forma sucinta e informal, algumas questões relevantes para a verificação das construções sintáticas da linguagem.

Uma declaração `data ID { ... }` define **um novo tipo de dados heterogênio** que é identificado pelo nome *ID*. O nome do novo tipo não deve conflitar com os nomes já existentes. O tipo de dados se comporta de forma idêntica a um *struct* da linguagem C, isto é, uma variável do tipo *ID* possui todos os campos declarados no tipo de dados. Os valores de tais campos podem ser acessados usando-se o operador `"."`.

Uma declaração de função funciona como uma função em C, exceto que em *lang*uma função pode retonhar mais de um valor. A linguagem deve suportar funções recursivas e sobrecarregadas e apenas os parâmetros da função devem ser usados para resolver a sobrecarga. O compilador deve assumir que os parâmetros da função são variáveis locais com os respectivos tipos declaradas. O compilador também deve se certificar de que todas as funções que retornam valores sempre retornam a quantidade e os tipos apropriados de valores.

Todo programa deve conter uma única função, designada *main* que não tem parâmetros e é função principal do programa.

Os comandos de *lang*são semelhantes aos comandos de linguagens de programação imperativas. o índice de vetores deve ter tipo inteiro. Operações aritméticas e lógicas seguem as regras usuais e não são válidas operações com operandos de tipos diferentes, e.g., *Float* e *Int*.

3 Entrega do Trabalho

A data da entrega do trabalho será até o dia **16 de novembro de 2020**.