1 Instruções Importantes

Nessa seoção apresentadas diversas informações relevantes referentes a entrega do trabalho e orientaçõees a serem seguidas durante a implementação do mesmo. Leia atentamente antes de começa-lo.

1.1 Equipe de Desenvolvimento

O trabalho será desenvolvido individualmente ou em dupla.

1.2 Escolha da Linguagem de Programação

O trabalho poderá ser desenvolvido em qualquer uma das linguagens a seguir, a seu critério: Java, Haskell, Racket, Rusty, C, C++.

1.3 Artefatos a Serem Entregues

Os artefatos a serem entregues são:

- código fonte do programa (Todo o projeto incluindo as ferramentas empregadas);
- arquivo de *makefile* para geração automática dos analisadores léxicos, sintéticos e compilação do fonte.
- documentação do trabalho em formato makrdow.

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, contendo nome e número matrícula do(s) autor(es) 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 testes do código entregue, análise do código-fonte e entrevista, que será presencial.

Os testes visam verificar a corretude do sistema desenvolvido. Para este propósito, instâncias de testes (programas $lang^2$) foram desenvolvidas especificamente para a fase de análise sintática e semântica. Na etapa de análise sintática, para ser considerado correto, o compilador desenvolvido deve aceitar todos os programas sintaticamente corretos e rejeitar todos os programas sintaticamente incorretos. Na etapa de análise semântica, o programa deve produzir os resultados esperados de acordo com as entradas fornecidas.

Os seguintes fatores serão observados na avaliação do código-fonte: estrutura do código, legibilidade e clareza. Os demais fatores avaliados no código-fonte são referentes à organização, escrita do trabalho e compreensão dos conteúdos relacionados.

A documentação do código deve conter informações relevantes para compilar, executar e auxiliar no entendimento do código-fonte. Ressalta-se que a documentação não deve conter cópias do código-fonte, pois este já é um dos artefatos enviados, mas deve apresentar as decisões de projeto tomadas: estruturas de

dados usadas para representação do programa, estratégia usada para interpretar a linguagem, dentre outras informações.

O trabalho deverá ser apresentado ao professor da disciplina e só será avaliado após a realização da entrevista, ou seja, será atribuída nota zero aos trabalhos que não forem apresentados. 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 o domínio dos autores sobre seu código tanto por meio de explicações sobre os pontos relevantes do trabalho desenvolvido, assim como pela realização de modificações no mesmo.

A distribuição dos pontos será dada conforme a Tabela 1. A pontuação dos testes semânticos foi subdividida em três categorias:

1. **simple:** Programas que não envolvem chamadas de funções, vetores ou tipos de dados definidos pelo usuário.
2. **function:** Programas que podem conter chamadas de função, mas não fazem uso de vetores ou tipos de dados definidos pelo usuário.
3. **full:** Programas que usam todos os recursos de $lang^2$.

Critério	Valor	Tipo
Testes Sintáticos	0,5	Binário
Testes Semânticos: simple	0,1	Binário
Testes Semânticos: function	0,2	Binário
Testes Semânticos: full	0,2	Binário
Entrevista	1,0	Contínuo
Exercício: Alteração do código	0,5	Contínuo

Tabela 1: Distribuição dos pontos entre os critérios

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:

Atrasos serão penalizados por uma função exponencial de dias de atraso, ou seja, será reduzido do percentual da nota a exponencial na base 2 dos dias de atraso. A Tabela 2 mostra a nota em função dos dias de atraso:

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

Tabela 2: Penalização por atraso na entrega

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

2 Especificação Técnica do Trabalho

Na sequência para construção do compilador da linguagem $lang^2$, a próxima etapa é verificar se um programa está correto em relação à especificação sintática (gramática) de $lang^2$e a construção de um interpretador para a linguagem. Neta primeira versão do interpretador as classes de tipos devem ser ignoradas. Qualquer uma das técnicas de análise sintática estudadas poderá ser empregada, assim como qualquer ferramenta de parser que implemente tais técnicas.

O compilador desenvolvido deverá oferecer uma interface de linha de comando e aceitar até dois parâmetros, sendo o primeiro uma das seguintes opções:

- "-syn" : Executa o analisador sintático e apenas imprime "accepted" se o programa estiver sintaticamente correto ou "rejected" caso contrário.
- "-i" : Executa o analisador sintático, constrói a AST e executa o interpretador sobre a AST construída.
- "-v" : Imprime a seguine mensagem " LangV2 - 2025/2 - v:0.1.2" Nas linhas seguintes imprime o número de matricula e o nome de cada membro da dupla, um por linha.

O comando "-v" não deve esperar um segundo parâmetro. Nos demais comandos o segundo parâmetro deve ser o caminho para o arquivo contendo o código-fonte a ser processado. Caso o compilador não receba nenhum parâmetro, ele deve imprimir uma mensagem de ajuda, informando quais parâmetros podem ser passados e em que ordem. Caso o compilador receba apenas um parâmetro, ele deverá ser considerado como o caminho do arquivo contendo o código-fonte. Caso o compilador receba dois parâmetros, então o primeiro é uma das opções supramencionadas e o segundo é o caminho do arquivo contendo o código-fonte. **Falha em obedecer esta interface poderá resultar em falha durante a execução dos testes.**

O trabalho deve ser compilado simplesmente invocando-se o comando make em seu diretório raiz. O resultado do make deve ser uma versão distributível do compilador (um executável se for em C, um jar se For em java etc...).

3 Entrega do Trabalho

A data da entrega do trabalho será até o dia **31 de Janeiro de 2025**. A entrega será realizada exclusivamente pela plataforma moodle.