Tema AGL, grupo agl-gg04

[TO DO]

- All agl vars must start with `__agl__` to avoid conflicts with python vars.
- Default values for agl vars.

- Join all versions of interpreter / semantic analysis and compiler in one branch.

Documentação extra

- Análise semântica - doc/semantic_check.md [POR COMPLETAR!!!]
- Exemplos AGL e xAGL - doc/examples [POR COMPLETAR!!!]
- Tutorial de execução - doc/running.md [POR COMPLETAR!!!]

Estrutura do repositório

O repositório está organizado da seguinte forma:

- `doc/` contém a documentação do projeto.
- `doc/examples/` contém exemplos de código AGL e xAGL.
- `doc/examples/demo/` contém videos de demonstração da linguagem AGL.
- `src/` contém o código fonte do projeto.
- `src/tests/` contém os testes do projeto (nomeadamente análise semântica).

Relatório

Introdução

Este documento apresenta o relatório do projeto final do grupo AGL-gg04 para a disciplina de Compiladores do ano letivo 2023/2024, focado na criação da linguagem de programação AG_L (***animated graphics language***).

- O objetivo geral deste trabalho foi o desenvolvimento de um ambiente de programação, constituído pela linguagem de programação AG_L (*animated graphics language*) e correspondentes ferramentas de compilação, que permite a criação de programas na linguagem de programação genérica *Python3*, usando a biblioteca *tkinter*, que, quando executados, permitem a visualização, com possível animação e interação, de *gráficos 2D*.

- A linguagem assume (implicitamente) a existência de uma área de desenho, *canvas*, com dimensões ilimitadas, sobre a qual se podem instanciar (desenhar) figuras gráficas 2D. Podem também ser instanciadas vistas (*View*), que permitem capturar o estado do *canvas* numa dada região, com um determinado zoom e num determinado instante no tempo. Há um conjunto

base de modelos gráficos pré-definidos, mas a linguagem possui mecanismos construtivos que permitem definir outros. As instâncias dos modelos gráficos podem, ao longo do tempo, mudar a sua posição e as suas propriedades, mas, apenas quando uma vista e *refrescada* é as alterações são capturadas por essa vista.

- Para além disso, a linguagem principal permite a importação de elementos auxiliares através de *descrições* numa linguagem secundária, cujos elementos, em *runtime*, possam ser carregados pelo programa final. Em concreto, foi definida uma linguagem secundária, aqui designada por xAG_L, associada a ficheiros com a extensão *xagl*, que permite auxilar a linguagem principal AG_L de alguma maneira (por exemplo, definindo modelos). O resultado desta tarefa é a gramática da linguagem xAG_L, a construção gramatical na linguagem AG_L que permite fazer a importação de descrições em xAG_L, o interpretador que permite fazer o parsing de descrições xAG_L, e exemplos ilustrando a sua utilização. Este interpretador foi desenvolvido na linguagem destino (neste caso *Python3*) e não na linguagem de trabalho (neste caso *Java*), uma vez que a interpretação é feita em *runtime*.

Torres de Hanoi: demonstra a facilidade na criação de programas complexos em AG_L. Utilizando multiplas vistas e modelos que facilitam a criação de animações complexas. Intro.gif

Requesitos do projeto

Foram definidos 4 níveis para a realização deste projeto:

Nível mínimo [DONE]

- Suporte para comentários, de linha e de bloco, tal como ilustrado no exemplos.
- Instanciação de uma vista (*View*) sobre a área de desenho (*canvas*). As vistas devem portam as ações de *move*, *refresh*, *wait* e *close*. Neste nível, considerámos que existe apenas uma vista.
- Instanciação dos modelos gráficos base *Dot*, *Line*, *Circle*, *Rectangle*, *Ellipse*, *Text*, *Arc*, *ArcChord* e *PieSlice*. Todos estes modelos têm implicitamente um ponto de referência, cuja localização no canvas será indicada aquando da sua instanciação. Todos os objetos gráficos devem suportar a ação de *move*. Todos os objetos gráficos possuem um conjunto de propriedades, que podem ser alteradas em tempo de execução.
- Suporte dos tipos de dados *Integer*, *Number*, *Point*, *Vector*, *String*, e *Time* e da instanciação de objetos destes tipos. Suporte de expressões envolvendo estes tipos de dados.
- Suporte da construção *with*.
- Uma construção gramatical repetitiva *for*, para iterar sobre uma sequência de valores.

- Verificação semântica no uso de variáveis e de expressões e na manipulação de propriedades.
- Definição e implementação de uma linguagem secundária, aqui designada por xAG_L, associada a ficheiros com a extensão *xagl*.

Nível desejável [DONE]

- Instanciação dos modelos gráficos base *Polyline*, *Spline*, *Polygon* e *Blob*. Todos estes modelos têm implicitamente um ponto de referência, cuja localização no *canvas* será indicada aquando da sua instanciação. Todos os objetos gráficos devem suportar a ação de *move*. Todos os objetos gráficos possuem um conjunto de propriedades, que podem ser alteradas em tempo de execução.
- O tipo de dados *Boolean* e a manipulação de expressões booleanas.
- Uma construção gramatical condicional. Neste nível, considerámos apenas a construção *if-else*.
- Uma construção gramatical repetitiva baseada em predicado (condição de termo ou de continuação). Neste nível, considerámos as construções *while*, *repeat-until*.
- Suporte para a especificação de vetores (*Vector*) em coordenadas polares.
- A possibilidade de definição de novos modelos (associada à palavra-chave *Model*), que permita agregar instâncias de outros modelos, base ou definidos anteriomente, numa nova entidade. Todas as propriedades destes novos modelos serão definidas aquando da sua conceção. Deve ser possível fazer depender as propriedades dos modelos incorporados de propriedades do modelo principal. O novo modelo poderá ser depois instanciado.
- Suporte para estrutura de dados iterável (array, lista, …) e de mecanismos de instanciação, acesso e manipulação dos seus elementos. Neste nível, considerámos apenas a construção de *array* (apenas aceita elementos do mesmo tipo).
- Possibilidade de aplicar as ações move, refresh e close diretamente a uma lista de objetos.

Nível adicional [DONE]

- Suporte para a rotação de objetos gráficos. Neste nível, houve a necessidade de redefinir internamente a forma como os objetos gráficos são representados (para permitir rotação de TODOS os objetos). Considerámos também a rotação de modelos que contêm outros modelos através do seu ponto de referência.
- Suporte para várias vistas.
- Suporte para a operação *deepcopy*, que permite a *cópia profunda* de objetos gráficos. Facilitação na duplicação de objetos gráficos complicados.

Desafios [DONE]

- Criação de uma animação interativa que permita a um utilizador resolver as torres de Hanoi.

Nível mínimo

Neste nível, o nosso grupo focou-se na implementação dos requisitos mínimos, tendo em conta a instânciação de uma vista, a instânciação dos modelos gráficos base, a manipulação de tipos de dados, a construção *with*, a construção *for*, a verificação semântica e a definição e implementação de uma linguagem secundária.

[POR COMPLETAR!!!]

Instanciação de uma vista

```
View v1 {
    move(100, 100)
    refresh()
    wait(1000)
    close()
}
```

Nível desejável

Neste nível, o nosso grupo focou-se na implementação dos requisitos desejáveis, tendo em conta a instânciação dos modelos gráficos base, a manipulação de tipos de dados, a construção *if-else*, a construção *while*, a construção *repeat-until*, a especificação de vetores (*Vector*), a definição de novos modelos, a estrutura de dados iterável e a aplicação das ações move, refresh e close diretamente a uma lista de objetos.

[POR COMPLETAR!!!]

Nível adicional

Neste nível, o nosso grupo focou-se na implementação dos requisitos adicionais, tendo em conta a rotação de objetos gráficos e a suporte para várias vistas.

[POR COMPLETAR!!!]

Nível extra

Neste nível, o nosso grupo focou-se na implementação dos requisitos extra, tendo em conta a criação de uma animação interativa que permita a um utilizador resolver as torres de Hanoi, a falicitação na criação de modelos predefinidos atribuindo valores default

a propriedades e a facilitação na duplicação de objetos gráficos complicados com a funcionalidade *deepcopy*.

[POR COMPLETAR!!!]

Constituição dos grupos e participação individual global

NMec	Nome	Participação
115637	GIOVANNI PEREIRA SANTOS	0.0%
113893	GUILHERME FERREIRA SANTOS	0.0%
104384	JOÃO PEDRO AZEVEDO PINTO	0.0%
114547	JOÃO PEDRO FERREIRA MONTEIRO	0.0%
113278	JORGE GUILHERME CONCEIÇÃO DOMINGUES	0.0%
115304	PEDRO MIGUEL AZEVEDO PINTO	0.0%

Contribuições

Para este trabalho, o nosso grupo dividiu-o nos seguintes tópicos e distribui os mesmos pelos elementos do grupo da seguinte forma:

[POR COMPLETAR!!!]

- Construção da gramática AGL:
  - Pedro Pinto - 115304
  - João Pinto - 104384
  - João Monteiro - 114547
- Compilador (AGL):
  - Pedro Pinto - 115304
  - Guilherme Santos - 113893
  - Giovanni Santos - 115637
- Análise semântica (AGL):
  - João Pinto - 104384
  - João Monteiro - 114547
  - Jorge Domingues - 113278
- Interpretador (xAGL):
  - Giovanni Santos - 115637
- Análise semântica (xAGL):
  - ...
- Testes:
  - Pedro Pinto - 115304
  - João Pinto - 104384
  - João Monteiro - 114547
  - Guilherme Santos - 113893
- Documentação:
  - Pedro Pinto - 115304

– ...