IPS Instituto Politécnico de Setúbal
Escola Superior de Tecnologia de Setúbal

PA 2019/20

Programação Avançada

Enunciado de Projeto
(Época Normal)

WebGraphs

A. Notas

- O presente enunciado descreve as regras, problema, requisitos obrigatórios, entrega e avaliação do projeto de Programação Avançada (PA) para a época normal.
- Leia atentamente o enunciado. Dúvidas sobre o seu conteúdo poderão ser colocados no fórum Moodle destinado ao projeto.
- Quaisquer casos omissos na descrição do problema e requisitos ficam ao critério dos alunos.

B. Regras

- O trabalho é **desenvolvido em grupos de dois alunos** do mesmo docente de laboratório. Qualquer desvio a esta regra tem de ser aceite pelo respetivo docente de laboratório antes da inscrição.
- Projetos não funcionais não serão avaliados, i.e., **um projeto que não permita** (usando uma interface gráfica) a visualização do "mapa" de um site em forma de grafo, a partir de um endereço dado**, é automaticamente reprovado.**
- A entrega é realizada através de submissão no Moodle.
- É obrigatória a discussão oral do trabalho. A falta à discussão oral resulta na classificação com 0 (zero).

B1. Conduta Académica

- **Ao submeter o projeto para avaliação, o(s) aluno(s) declara(m), sob compromisso de honra, que o projeto desenvolvido é original e foi desenvolvido pelos mesmos.**
- Projetos copiados total ou parcialmente serão automaticamente cotados com 0 (zero), sem averiguação de qual é o original e qual é a cópia.
  - Todos os projetos serão submetidos à plataforma MOSS (https://theory.stanford.edu/~aiken/moss/) para deteção de cópias.

0. Introdução

O projeto consiste no desenvolvimento de uma aplicação de *desktop* para geração e visualização de porções da WWW na forma de grafos.

A aplicação tem de ser desenvolvida utilizando TADs, padrões de software e interface gráfica em JavaFX.

A secção **I** explica o objetivo geral da aplicação e na secção II são apresentados os requisitos funcionais e não-funcionais que a aplicação deverá respeitar. A secção **III** informa sobre a data de entrega do projeto e sobre o conteúdo do relatório. Finalmente a secção **IV** contém a tabela de cotações que será aplicada aos projetos submetidos.

I. Problema

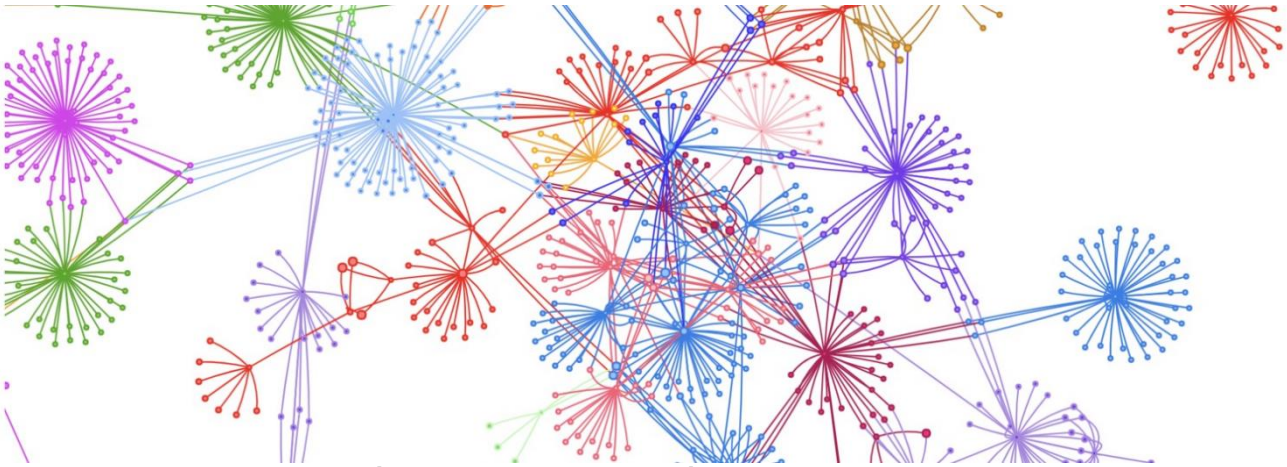

Figura 1 - Representação gráfica de um conjunto de páginas web.

Pretende-se desenvolver uma aplicação que represente uma porção da WWW na forma de grafo, i.e., páginas web na forma de vértices e links na forma de arestas (ver Figura 1).

A aplicação deve permitir inspecionar a estrutura do grafo e, através deste, obter diversas estatísticas entre as respetivas páginas.

II. Requisitos

Os seguintes requisitos funcionais e não-funcionais são os requeridos na resolução do problema.

1 Geração de modelos (web crawling)

A aplicação deve assentar num *web crawler* que, dado o endereço de uma página inicial, irá construir a representação em grafo dessa página, das páginas referenciadas por essa (através de hiperligações) e subsequentemente para cada página encontrada -- este grafo será doravante denominado por **modelo**.

Os vértices deverão ser etiquetados com o título da página e as arestas com o texto da hiperligação.

No decurso da geração do modelo poderão ser encontradas hiperligações para páginas inexistentes. Estas devem ser representadas por um vértice etiquetado com "404 – Not Found". A cada página distinta não encontrada deve corresponder um vértice distinto.

A geração dos modelos deve poder funcionar em dois modos:

A. Automático
B. Iterativo

A. Modo automático

No modo automático, a partir de uma página inicial, a aplicação executa uma visita automática de páginas até que seja satisfeito um determinado critério de paragem.

Independentemente do critério, a geração do modelo deve assentar numa visita em largura (*breadth-first search*), e.g., todas as páginas referenciadas pela página inicial devem ser visitadas antes de quaisquer outras.

Critérios de paragem

Os critérios de paragem para visitas de páginas são:

- Páginas visitadas – termina assim que N páginas forem visitadas;
- Profundidade – termina assim que todos as páginas que "distam" M links da página inicial forem visitadas.

B. Modo iterativo

No modo iterativo é o utilizador que determina o ritmo e ordem de visita das páginas e, subsequentemente, da geração do modelo e sua visualização.

Dada uma página inicial, deverá ser inicializado o modelo que representa essa página e apresentada uma lista *L* das hiperligações nela contidas. Da lista *L* deve ser possível escolher uma hiperligação a partir do qual o processo itera mais uma vez.

A qualquer instante o utilizador pode selecionar uma página já visitada, i.e., um vértice do modelo, para continuar o processo.

Undo

Neste processo iterativo deve ser possível a qualquer instante o utilizador reverter a visita da última página, sendo esta ação refletida no modelo. Esta ação pode continuar até terem sido revertidas todas as visitas, exceto a inicial.

Isto corresponde a uma funcionalidade de "undo multi-nível".

2 Visualização / Interação

1. Deve ser possível visualizar o modelo em tempo-real, i.e., à medida que as páginas são visitadas, tendo em atenção que:
   - O vértice da página inicial deve ter uma cor distinta dos restantes;
   - Vértices correspondentes a páginas não encontrados devem ser vermelhos.
2. Após um modelo ter sido gerado deve haver a possibilidade de selecionar uma página no modelo e abri-la no browser do computador.

3 Persistência

Deve ser possível **exportar** (guardar em ficheiro) um modelo gerado por qualquer um dos modos de crawling. Os tipos de ficheiro a suportar devem ser:

1. *java persistence (serialization)*
2. formato JSON.

Deve então ser também possível **importar** um modelo previamente guardado, em qualquer um dos dois formatos.

4 Estatísticas

Sobre um modelo gerado deve ser possível consultar a seguinte informação:

1. Páginas visitadas (número e listagem);
2. Contagem de páginas distintas não encontradas;
3. Página mais referenciada por outras;
4. Outro indicador que achar relevante;
5. Gráfico de barras com o número de ligações das 5 páginas com mais hiperligações;
6. Outro gráfico que achar relevante.

5 Visualização do caminho mais curto

Deve ser possível visualizar em forma de árvore o caminho mais curto entre a página inicial e cada uma das páginas descobertas.

6 Logging

Deve ser mantido um log do processo de geração de um modelo. Este log é reiniciado a cada nova geração. Cada entrada do log deve ter o seguinte formato:

```
<timestamp> | <título_página> | <endereço_http> | <título_página_origem> | <numero_hiperligacoes_pagina>
```

7 Padrões de Software

A utilização de padrões de software é essencial no desenvolvimento e cumprimento dos requisitos funcionais anteriores.

É esperado que se utilizem alguns dos padrões de software lecionados na unidade curricular, sendo a sua aplicabilidade a cada requisito funcional da competência do(s) aluno(s) programador(es).

8 Javadoc

O código deverá ser documentado com recurso ao Javadoc, precedendo todo e qualquer método da descrição do seu propósito. Comentários adicionais deverão ser colocados em trechos de código onde a compreensão por parte de terceiros possa ser menos óbvia.

III. Entrega e Discussões

O projeto obedece a uma **milestone** intermédia (15%) e a uma entrega **final** em duas fases (85%) - versão funcional e após *refactoring* de bad-smells detetados na versão anterior. **Após a penúltima submissão (versão funcional) não podem ser adicionadas novas funcionalidades nem correções às funcionalidades existentes.**

Chama-se a atenção o seguinte:

- A não entrega da **milestone** não impossibilita as restantes entregas, mas implica a perda total da pontuação referente à mesma (15% da nota total);
- **Se for entregue um projeto não-funcional (um projeto que não permita** a visualização de um modelo na forma de grafo, a partir de um endereço dado**), o projeto é reprovado e não é aceite a entrega final;**
- As entregas serão penalizadas com 0,5 (meio) valor por cada hora de atraso. A primeira hora é considerada 15min após a hora limite (**09:00**), hora do servidor;
- As discussões são obrigatórias para todos os elementos do grupo de trabalho;
- O conteúdo das entregas correspondem aos respetivos projetos **Netbeans** desenvolvidos + respetivo relatório em PDF, contidos num ficheiro ZIP/RAR/7z e cujo nome obedeça ao seguinte formato: **<num1>__<num2>.{zip, rar, 7z}**.

Entrega Milestone via Moodle: 25 de novembro (segunda-feira) de 2019. Apresentações presenciais a agendar posteriormente.

Entrega Pré-final via Moodle: 6 de janeiro (segunda-feira) de 2019. Apresentações presenciais a agendar posteriormente.

Entrega Final via Moodle: 23 de janeiro (quarta-feira) de 2019. Discussões a serem agendadas entre 25 e 29 de Janeiro.

Para as entregas existirão links de submissão relativos a cada turma de laboratório. Em algum caso onde membros do grupo pertençam a turmas diferentes, a submissão deverá ser feita em apenas um dos links.

1 Milestone

A milestone (MS) corresponde à entrega de:

Um projeto NetBeans contendo:

1. Implementação de um TAD DiGraph (variante das apresentadas nas aulas TP e que a interface base será disponibilizada);
2. Classe **WebCrawler** que inclua uma instância da implementação anterior e que contenha métodos responsáveis por criar um modelo no modo automático segundo o critério: número máximo de páginas visitadas, e mostrar no ecrã o grafo (pode ser em modo texto) e pelo menos 3 indicadores estatísticos (ver secção 4).
3. Testes JUnit para as classes implementadas nos pontos 1 e 2.

Um mini-relatório contendo:

1. O pseudocódigo dos algoritmos de geração dos modelos em:
   - em modo iterativo, e;
   - em modo automático.

2. Uma mockup da interface gráfica que será desenvolvida (não é vinculativa para a fase seguinte).

2 Entrega Final

A entrega final é efetuada em duas fases. Cada fase tem uma data de submissão distinta e correspondentes discussões.

Versão Pré-Final (funcional)

Corresponde à submissão de uma aplicação JavaFX que resolva o problema proposto em (I) e que obedeça aos requisitos estabelecidos em (II).

Projetos não-funcionais, não serão avaliados.

Versão Final

A versão final é composta por o (i) projeto NetBeans e (ii) o Relatório.
O projeto NetBeans corresponde à aplicação submetida anteriormente após:

1. a implementação de um novo requisito funcional (a ser disponibilizado no dia 7 de janeiro),
2. à melhoria da estrutura do código após ter sido realizado *refactoring ao mesmo*.

A adição de outras novas funcionalidades e/ou correção de problemas anteriores não serão contabilizadas.

O relatório deverá conter:

1. A especificação de TADs implementados;
2. Diagrama(s) de classes (diagramas ilegíveis serão contabilizados com 0 valores);
3. Documentação de todas as classes (sugere-se reutilizar a documentação Javadoc);
4. Descrição/uso/justificação dos padrões de software utilizados
   - deverá indicar para cada um dos padrões utilizados, a descrição do mesmo, qual a razão de esse ter sido selecionado e de que forma as classes do padrão foram mapeadas para a solução concreta;
5. Refactoring:
   - Uma tabela com os tipos de *bad-smells* detetados, quantas situações deste tipo foram detetadas e técnica de *refactoring* aplicada para a sua correção.
   - Um exemplo de cada correção efetuada, com código antes e após *refactoring*.

IV. Critérios de Avaliação

A aplicação deverá ser implementada na linguagem Java segundo as boas práticas de POO, utilizando os Tipos Abstratos de Dados (TAD) lecionados e os Padrões de Software relevantes.

É fornecida, a título de suporte, a tabela de cotações do projeto. A cotação de cada item será ponderada pela qualidade da solução/código obtida.

Fase		Item	Cotação/ Penalização
Milestone (15%)		Implementação do TAD DiGraph	30
		Classe WebCrawler – Construção do Grafo	40
		Classe WebCrawler – Métodos para cálculo das estatísticas	40
		Testes Junit	60
		Mini-Relatório	20
		Javadoc	10
		TOTAL	**200**
Entrega Final (85%)	**Funcionalidades**	Modelo – Modo Automático	15
		Critérios de Paragem	10
		Modelo – Modo Iterativo	15
		Undo	10
		Logger	10
		Persistência	15
		Estatísticas	10
		Árvore Caminho mais curto	20
		Interface Gráfica (Usabilidade geral)	10
		Visualização do modelo (grafo) com diferentes cores	10
		Interação na construção do modelo em modo iterativo	10
		Funcionalidade Extra	10
	Javadoc		15
	Relatório		**40**
	TOTAL		**200**
	Existência de bad smells (por cada não tratado)		-5 (até -30)
	Não utilização de padrões de software adequados (por cada situação)		-15 (até -60)

(fim de enunciado)