Enunciado

Laboratórios de Informática II

Laboratórios de Programação II

1 Requisitos

Requisitos para todas as entregas

Requisitos principais

- Funcionalidade;
- Qualidade de software;
- Reutilização de código.

Requisitos para todas as entregas

Projeto

- Deve ser desenvolvido em C;
- Deve executar na máquina virtual disponibilizada no endereço `lambda.di.uminho.pt`;
- Todas as funções **não podem ultrapassar** a **complexidade ciclomática** de 10;
- Todas as funções **não podem ultrapassar** as 15 instruções;
- O código deve ser **modular**;
- Deve existir uma separação entre a **lógica** e o **interface**;
- Todas as funções da lógica **têm que ser** testadas com **`cunit`**;
- Todas as funções **têm que ser** documentadas.

Avaliação

Etapa	Percentagem	Prazo	Avaliação
1	25%	22 de Março	23 a 28 de Março
2	25%	19 de Abril	20 a 25 de Abril
3	25%	24 de Maio	25 a 30 de Maio
Final	25%	31 de Maio	1 a 5 de Junho

Avaliação

Penalização

- Se um projeto não cumprir qualquer dos requisitos, este não será avaliado
- Se nenhum aluno for capaz de responder a uma pergunta sobre o código, a avaliação do projeto será **NULA**
- Os docentes reservam-se o direito de chamar os alunos para uma nova defesa oral, individualmente ou em grupo, caso surja alguma dúvida sobre o projeto
- Caso os alunos se recusem a responder a qualquer pergunta, ou se recusem a comparecer, serão reprovados à UC

Avaliação Individual

- O não comparecimento às defesas das etapas implica a *não avaliação* (**ZERO**) nessa etapa;
- O não ser capaz de responder a qualquer pergunta numa etapa implica o mesmo que a alínea anterior;
- A avaliação de cada elemento deverá ser uma **percentagem** da avaliação do projeto.

Avaliação de pares

- Cada elemento deverá avaliar os seus colegas em cada etapa;
- Caso um elemento não avalie os seus colegas, ele terá **ZERO** nessa etapa.

2 Etapas

Golf

Jogo

- Exemplo
- 7 pilhas de 5 cartas são dispostas;
- A pilha de descarte começa com uma carta visível;
- Pode-se pegar numa carta do topo da pilha e colocar na pilha de descarte desde que seja o valor imediatamente acima ou abaixo do que está no topo da pilha de descarte;
- A qualquer altura pode-se biscar uma carta do baralho e colocar no topo da pilha de descarte.

Implementação

- Command Line Interface (CLI)
- Utilização de UTF-8 para mostrar as cartas
- Crie os módulos e estruturas de dados que servirão de base para o projeto

Simple Simon

Jogo

- Exemplo
- Layout de colunas;
- O foco é a movimentação de sequências de cartas entre colunas e a gestão de espaços vazios;

Implementação

- Deverá reutilizar o **máximo** de lógica de manipulação de cartas desenvolvida na etapa anterior.

Terceira etapa

Objectivo

- Implementar um sistema que leia um ficheiro com uma linguagem de descrição de uma paciência e a implemente;
- Deve permitir jogar com dicas e *undo* tal como os exemplos;
- Deve permitir resolver a paciência
- A linguagem só vai ser especificada mais tarde!

Requisitos obrigatórios

- Deverá reutilizar o **máximo** de lógica de manipulação de cartas desenvolvida nas etapas anteriores.

3 Ferramentas

pmccabe

Vamos utilizar:

- *Modified McCabe Cyclomatic Complexity*
- *Statements in function*

```
rcm@lambda:~/guioes/G1$ pmccabe -v *.c
Modified McCabe Cyclomatic Complexity
|   Traditional McCabe Cyclomatic Complexity
|       |    # Statements in function
|       |        |   First line of function
|       |        |       |   # lines in function
|       |        |       |       |  filename(definition line number):function
|       |        |       |       |           |
1 1 3 5 7 main.c(5): main
1 1 4 4 7 quad.c(4): raizes
```