



Engenharia Informática

Sistemas Operativos

Relatório

Trabalho Prático – Meta 1

Ano Letivo: 2020/2021

Curso: Engenharia Informática

Autores: Ricardo Pereira – 2015015815, Pedro Ruivo - 2015016289

Disciplina: Sistemas Operativos

Turma: P1

Professores: João Durães, Jorge Rodrigues

Data de Submissão: 08/11/2020

Índice

1. Referee (Árbitro).....	3
1.1. Constantes	3
1.2. Estruturas de dados	3
2. Player (Jogador)	4
2.1. Estruturas de dados	4
3. RNGGuess (Jogo)	5
4. Utils e Makefile.....	6

1. Referee (Árbitro)

Na meta 1, o árbitro apenas prepara a sua aplicação. Onde absorve os parâmetros recebidos pela consola e armazena na sua estrutura de dados, esta funcionalidade não está totalmente testada, mas foi desenvolvida de modo a que a ordem não seja importante, mas a aplicação será encerrada (corretamente) se os parâmetros não forem fornecidos como esperado. Juntamente com os parâmetros, esta aplicação também obtém variáveis de ambiente. Se não conseguir obter as variáveis de ambiente, esta irá usar os valores pré-definidos no seu ficheiro *header*.

1.1. Constantes

```
#define DEFAULT_GAMEDIR "~/Documents/SO/SO_CHAMPION/Games/"
#define DEFAULT_MAXPLAYER 10

#define MIN_CHAMP_DURATION 60000
#define MAX_CHAMP_DURATION 600000

#define MIN_WAITING_DURATION 30000
#define MAX_WAITING_DURATION 120000

#define MAX_MAXPLAYER 30
```

Estas constantes são autoexplicativas e usadas para evitar falhas no uso da aplicação.

1.2. Estruturas de dados

```
typedef struct Referee Referee;
typedef struct Application Application;

struct Referee{
    char gameDir[STRING_MEDIUM];
    int maxPlayers;

    int championshipDuration;
    int waitingDuration;
};

struct Application{
    Referee referee;
};
```

A estrutura de dados principal, *Application* é usada para armazenar a informação necessária para o funcionamento da aplicação. Esta contém uma outra estrutura, *Referee*, que armazena a informação geral do árbitro.

2. Player (Jogador)

O jogador é a aplicação que não tem muito desenvolvimento na meta 1. Esta aplicação apenas prepara a sua aplicação e pede o nome de utilizador ao jogador.

2.1. Estruturas de dados

```
typedef struct Player Player;

typedef struct Application Application;

struct Player{
    char username[STRING_MEDIUM];
};

struct Application{
    Player player;
};
```

A estrutura de dados principal, *Application* é usada para armazenar a informação necessária para o funcionamento da aplicação. Esta contém uma outra estrutura, *Player*, que armazena a informação geral do jogador.

3. RNGGuess (Jogo)

Para o primeiro jogo, foi desenvolvido um simples jogo de adivinhar um número randomizado. Este jogo tem uma pequena introdução, desenvolvimento, e armazena a pontuação do jogador ao longo do jogo. Esta aplicação atualmente não tem qualquer constante nem estruturas de dados.

4. Utils e Makefile

Adicionalmente, foi criado um ficheiro com o objetivo de ser uma fonte de funcionalidades úteis para todas as aplicações. Com este ficheiro, o desenvolvimento do projeto deverá ser mais dinâmico, simples e espera-se evitar bugs simples.

O *makefile* compila todo o código e com as dependências necessárias. Os seguintes comandos são possíveis:

- Make – executa a primeira *tag*;
- Make full – compila e executa;
- Make cleanObj – remove os ficheiros compilados;
- Os restantes comandos podem ser chamados, mas não são necessários, visto que os de cima executam tudo o que é necessário.