Sistemas Operativos

Ano Letivo: 2020/2021

Tomé Lopes Carvalho - 97939

Gonçalo Fernandes Machado - 98359

Trabalho 2

Simulação de jogo de futebol

Nota: O termo “jogador” significa “player ou goalie”.

**Descrição geral da abordagem**

**Referee**

1. Chegar ao local do jogo, demorando um tempo aleatório
2. Esperar que as duas equipas estejam formadas
3. Dar início à partida
4. Arbitrar durante um tempo aleatório
5. Terminar a partida

**Player**

1. Chegar ao local do jogo, demorando um tempo aleatório
2. Verificar se chegou atrasado (ou seja, se já estão 2 \* NUMTEAMPLAYERS players no campo)
   1. Se tiver chegado atrasado, não participará no jogo
   2. Se não tiver chegado atrasado, verificar se pode formar uma equipa com os jogadores presentes no campo no instante em que chega
      1. Se puder formar uma equipa, aceitar para a sua equipa jogadores no campo que estejam à espera de serem escolhidos. Informar o referee de que foi formada a equipa
      2. Se não puder formar uma equipa, ficar à espera de ser escolhido para uma equipa formada por um jogador que virá mais tarde
      3. Começar a jogar quando o referee sinalizar o início do jogo
      4. Jogar até o referee sinalizar o fim do jogo

**Goalie**

1. Chegar ao local do jogo, demorando um tempo aleatório
2. Verificar se chegou atrasado (ou seja, se já estão 2 \* NUMTEAMGOALIES goalies no campo)
   1. Se tiver chegado atrasado, não participará no jogo
   2. Se não tiver chegado atrasado, verificar se pode formar uma equipa com os jogadores presentes no campo no instante em que chega
      1. Se puder formar uma equipa, aceitar para a sua equipa jogadores no campo que estejam à espera de serem escolhidos. Informar o referee de que foi formada a equipa
      2. Se não puder formar uma equipa, ficar à espera de ser escolhido para uma equipa formada por um jogador que virá mais tarde
      3. Começar a jogar quando o referee sinalizar o início do jogo
      4. Jogar até o referee sinalizar o fim do jogo

**Funcionamento de cada programa**

**Referee**

1. arrive()
   1. Entrar na região crítica;
   2. Mudar o estado do referee para ARRIVING (0);
   3. Guardar estado no log file;
   4. Sair da zona crítica;
   5. Esperar um tempo aleatório;
2. waitForTeams()
   1. Entrar na região crítica;
   2. Mudar o estado do referee para WAITING\_TEAMS (1);
   3. Guardar estado no log file;
   4. Sair da zona crítica;
   5. Fazer *semDown* do semáforo *refereeWaitTeams* 2 vezes, 1 vez por cada equipa, ou seja esperar que os jogadores que formam as equipas notifiquem que as suas equipas estão prontas.
3. startGame()
   1. Entrar na região crítica;
   2. Mudar o estado do referee para STARTING\_GAME (2);
   3. Guardar estado no log file;
   4. Sair da zona crítica;
   5. Fazer *semUp* do semáforo *playersWaitReferee* para todos os membros de ambas as equipas, que neste caso são 4 players e 1 goalie de cada equipa. Este passo serve como uma notificação aos jogadores que o jogo está a começar;
4. play()
   1. Entrar na região crítica;
   2. Mudar o estado do referee para REFEREEING (3);
   3. Guardar estado no log file;
   4. Sair da zona crítica;
   5. Esperar um tempo aleatório para terminar o jogo;
5. endGame()
   1. Entrar na região crítica;
   2. Mudar o estado do referee para ENDING\_GAME (4);
   3. Guardar estado no log file;
   4. Sair da zona crítica;
   5. Fazer *semUp* do semáforo *playersWaitEnd* para todos os membros de ambas as equipas. Este passo é para simular o apito final do jogo, onde o referee avisa os jogadores que o jogo acabou;

**Player**

1. arrive(*id* do player)
   1. Entrar na região crítica;
   2. Mudar o estado do player para ARRIVING (0);
   3. Guardar estado no log file;
   4. Sair da zona crítica;
   5. Esperar algum tempo;
2. playerConstituteTeam(*id* do player)
   1. Entrar na região crítica;
   2. Incrementar *playersArrived*, para indicar que o player chegou;
   3. Incrementar *playersFree*, para indicar que o player está livre;
   4. Comparar *playersArrive* com 2 \* NUMTEAMPLAYERS, ou seja, comparar o número de players que já chegou com o número de players que é preciso para o jogo. Caso *playersArrive* seja maior(player está atrasado):
      1. Mudar o estado do player para LATE (7);
      2. Guardar estado no log file;
      3. Ativar a flag *late*, que será usada fora da zona crítica;
   5. Caso *playersArrive* seja menor (player chegou a tempo):
      1. Verificar se existem jogadores suficientes para formar equipa. Caso se possa formar equipa:
         1. Subtrair NUMTEAMPLAYERS a *playersFree*, ou seja, dizer que deixam de estar NUMTEAMPLAYERS players livres, pois irão entrar numa equipa;
         2. Subtrair NUMTEAMGOALIES a *goaliesFree*, ou seja, dizer que deixam de estar NUMTEAMGOALIES goalies livres, pois irão entrar numa equipa;
         3. Mudar o estado do player para FORMING\_TEAM (2);
         4. Guardar estado no log file;
         5. Ativar a flag *forming*, que será usada fora da zona crítica;
         6. Fazer *semUp* do semáforo *goaliesWaitTeam* NUMTEAMGOALIES vezes, ou seja, chamar os goalies que irão pertencer à equipa;
         7. Fazer *semUp* do semáforo *playersWaitTeam* NUMTEAMPLAYERS - 1 vezes, ou seja, chamar os players que irão pertencer à equipa exceto o próprio player;
         8. Fazer *semDown* do semáforo *playersRegistered* NUMTEAMPLAYERS - 1 + NUMTEAMGOALIES vezes, ou seja, esperar que todos os jogadores da equipa menos o próprio player digam que entenderam que estão na equipa;
         9. Igualar ret a *teamId* e incrementar *teamId*, para o próximo jogador que formar uma equipa;
         10. Fazer *semUp* do semáforo *refereeWaitTeam*, ou seja, notificar o referee que a equipa está pronta;
      2. Caso o player não possa formar equipa:
         1. Mudar o estado do player para WAITING\_TEAM (1);
         2. Guardar estado no log file;
   6. Sair da zona crítica;
   7. Verificar se as flags *late* ou *forming* foram ativadas. Caso não tenham sido, ou seja, o player está à espera de ser chamado para uma equipa:
      1. Fazer *semDown* do semáforo *playersWaitTeam*, ou seja, esperar até ser chamado para uma equipa;
      2. Igualar *ret* a *teamId*;
      3. Fazer *semUp* do semáforo *playersRegistered*, ou seja, notificar o jogador que formou a equipa que o player entendeu que está na equipa;
   8. Retornar *ret*, ou seja, a equipa onde o player ficou;
3. waitReferee(*id* do player, equipa do player)
   1. Entrar na região crítica;
   2. Mudar o estado do player para WAITING\_START\_1 (3) caso a equipa seja 1, ou para WAITING\_START\_2 (4), caso a equipa seja 2;
   3. Guardar estado no log file;
   4. Sair da zona crítica;
   5. Fazer *semDown* do semáforo *playersWaitReferee*, ou seja, esperar que o referee notifique o início do jogo;
4. playUntilEnd(*id* do player, equipa do player)
   1. Entrar na região crítica;
   2. Mudar o estado do player para PLAYING\_1 (3) caso a equipa seja 1, ou para PLAYING\_2 (4), caso a equipa seja 2;
   3. Guardar estado no log file;
   4. Sair da zona crítica;
   5. Fazer *semDown* do semáforo *playersWaitEnd*, ou seja, esperar que o referee notifique o fim da partida;

**Goalie**

1. arrive(*id* do goalie)
   1. Entrar na região crítica;
   2. Mudar o estado do goalie para ARRIVING (0);
   3. Guardar estado no log file;
   4. Sair da zona crítica;
   5. Esperar algum tempo;
2. playerConstituteTeam(*id* do goalie)
   1. Entrar na região crítica;
   2. Incrementar *goaliesArrived*, para indicar que o goalie chegou;
   3. Incrementar *goaliesFree*, para indicar que o goalie está livre;
   4. Comparar *goaliesArrive* com 2 \* NUMTEAMGOALIES, ou seja, comparar o número de goalies que já chegou com o número de goalies que é preciso para o jogo. Caso *goaliesArrive* seja maior(goalie está atrasado):
      1. Mudar o estado do player para LATE (7);
      2. Guardar estado no log file;
      3. Ativar a flag *late*, que será usada fora da zona crítica;
      4. Caso *goaliesArrive* seja menor (goalies chegou a tempo):
      5. Verificar se existem jogadores suficientes para formar equipa. Caso se possa formar equipa:
         1. Subtrair NUMTEAMPLAYERS a *playersFree*, ou seja, dizer que deixam de estar NUMTEAMPLAYERS players livres, pois irão entrar numa equipa;
         2. Subtrair NUMTEAMGOALIES a *goaliesFree*, ou seja, dizer que deixam de estar NUMTEAMGOALIES goalies livres, pois irão entrar numa equipa;
         3. Mudar o estado do goalie para FORMING\_TEAM (2);
         4. Guardar estado no log file;
         5. Ativar a flag *forming*, que será usada fora da zona crítica;
         6. Fazer *semUp* do semáforo *goaliesWaitTeam* NUMTEAMGOALIES - 1 vezes, ou seja, chamar os goalies que irão pertencer à equipa, exceto o próprio goalie;
         7. Fazer *semUp* do semáforo *playersWaitTeam* NUMTEAMPLAYERS vezes, ou seja, chamar os players que irão pertencer à equipa;
         8. Fazer *semDown* do semáforo *playersRegistered* NUMTEAMPLAYERS - 1 + NUMTEAMGOALIES vezes, ou seja, esperar que todos os jogadores da equipa menos o próprio jogador digam que entenderam que estão na equipa;
         9. Igualar *ret* a *teamId* e incrementar *teamId*, para o próximo jogador que formar uma equipa;
         10. Fazer *semUp* do semáforo *refereeWaitTeam*, ou seja, notificar o referee que a equipa está pronta;
      6. Caso o jogador não possa formar equipa:
         1. Mudar o estado do goalie para WAITING\_TEAM (1);
         2. Guardar estado no log file;
   5. Sair da zona crítica;
   6. Verificar se as flags *late* ou *forming* foram ativadas. Caso não tenham sido, ou seja, o goalie está à espera de ser chamado para uma equipa:
      1. Fazer *semDown* do semáforo *playersWaitTeam*, ou seja, esperar até ser chamado para uma equipa;
      2. Igualar ret a teamId;
      3. Fazer *semUp* do semáforo *playersRegistered*, ou seja, notificar o jogador que formou a equipa que o goalie entendeu que está na equipa;
   7. Retornar *ret*, ou seja, a equipa onde o jogador ficou;
3. waitReferee(*id* do goalie, equipa do goalie)
   1. Entrar na região crítica;
   2. Mudar o estado do goalie para WAITING\_START\_1 (3) caso a equipa seja 1, ou para WAITING\_START\_2 (4), caso a equipa seja 2;
   3. Guardar estado no log file;
   4. Sair da zona crítica;
   5. Fazer *semDown* do semáforo *playersWaitReferee*, ou seja, esperar que o referee notifique o início do jogo;
4. playUntilEnd(*id* do goalie, equipa do goalie)
   1. Entrar na região crítica;
   2. Mudar o estado do goalie para PLAYING\_1 (3) caso a equipa seja 1, ou para PLAYING\_2 (4), caso a equipa seja 2;
   3. Guardar estado no log file;
   4. Sair da zona crítica;
   5. Fazer *semDown* do semáforo *playersWaitEnd*, ou seja, esperar que o referee notifique o fim da partida;

**Tabela de semáforos**

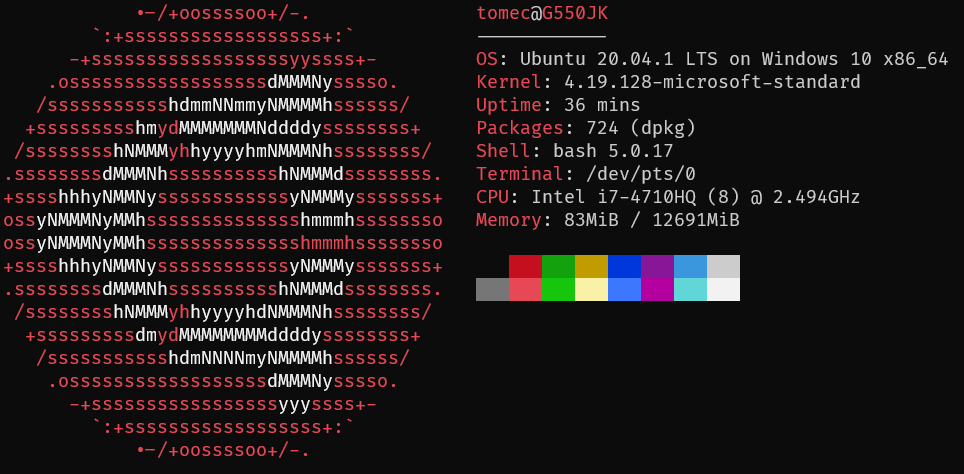
| Semáforo | Entidade down | Função down | Nº de downs | Entidade up | Função up | Nº de ups |
| --- | --- | --- | --- | --- | --- | --- |
| playersWaitTeam | player | playerConstituteTeam | 6/7/8\***1** | player, goalie | playerConstituteTeam | 6/7/8\***1** |
| goaliesWaitTeam | goalie | playerConstituteTeam | 0/1/2\***2** | player, goalie | playerConstituteTeam | 0/1/2\***2** |
| playersWaitReferee | player, goalie | waitReferee | 8 players,  2 goalies | referee | startGame | 10 |
| playersWaitEnd | player, goalie | playUntilEnd | 8 players,  2 goalies | referee | endGame | 10 |
| refereeWaitTeams | referee | waitForTeams | 2 | player, goalie | playerConstituteTeam | 2 |
| playerRegistered | player, goalie | playerConstituteTeam | 8 players,  2 goalies | player, goalie | playerConstituteTeam | 8 players,  2 goalies |

\***1** - 6 se 2 players formarem equipas, 7 se um player formar uma equipa, 8 se dois players formarem uma equipa

\***2** - 0 se 2 goalies formarem equipas, 1 se um goalie formar uma equipa, 2 se nenhum goalie formar uma equipa

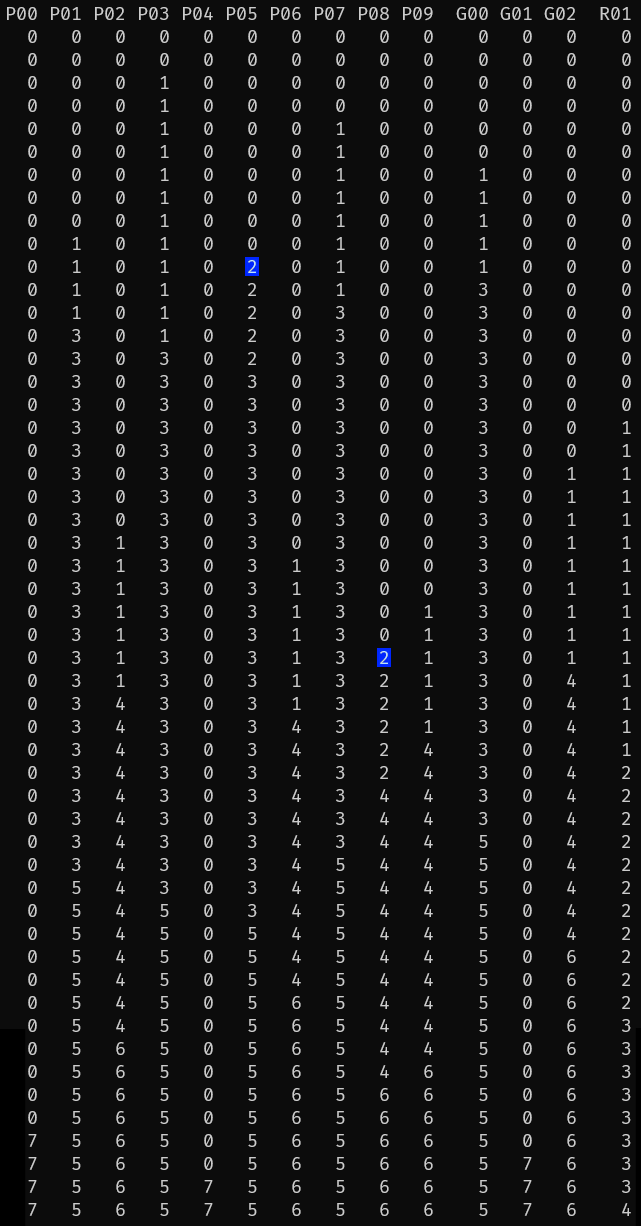
**Testes**

Realizados em Ubuntu 20.04 a correr através de WSL2 no Windows 10.



Estão incluídos no ficheiro zip do código fonte ficheiros de texto com as execuções do programa seguintes (0 corresponde a “nosso” e 1 a “professor”):

| **Player** | **Goalie** | **Referee** |
| --- | --- | --- |
| Nosso | Nosso | Nosso |
| Nosso | Nosso | Professor |
| Nosso | Professor | Nosso |
| Nosso | Professor | Professor |
| Professor | Nosso | Nosso |
| Professor | Nosso | Professor |
| Professor | Professor | Nosso |



Para podermos considerar que o programa está completamente funcional o programa precisa de cumprir os seguintes critérios:

-Player/Goalie:

-Inicia com estado 0 (ARRIVING);

-Quando, ao chegar, já existem pessoas suficientes para as duas equipas, alterar o seu estado para 7 (LATE) e terminar;

-Quando não existem pessoas suficientes para formar equipa mas ainda não

existem players suficientes para 2 equipas , alterar o seu estado para 1

(WAITING\_TEAM);

-Quando existem pessoas suficientes para formar equipa, alterar o

seu estado para 2 (FORMING\_TEAM);

-Cada jogador muda para o estado 3 (WAITING\_START\_1) ou 4

(WAITING\_START\_2)

consoante a sua equipa atribuída;

-O jogador que cria a equipa é o último a trocar o seu estado para 3 ou 4;

-Quando o referee mudar o estado para 2 (STARTING\_GAME), trocar o estado

para 5 (PLAYING\_1) ou 6 (PLAYING\_2) consoante a equipa;

-Quando o referee muda para o estado 4, terminar;

-Referee:

- Inicia com estado 0 (ARRIVING);

- Altera o estado para 1 (WAITING\_TEAMS);

- Quando as duas equipas estiverem formadas, mudar estado para

2 (STARTING\_GAME);

- Quando estiverem quase todos/todos os jogadores com estado 5 ou 6, mudar

para estado 3 (REFEREEING);

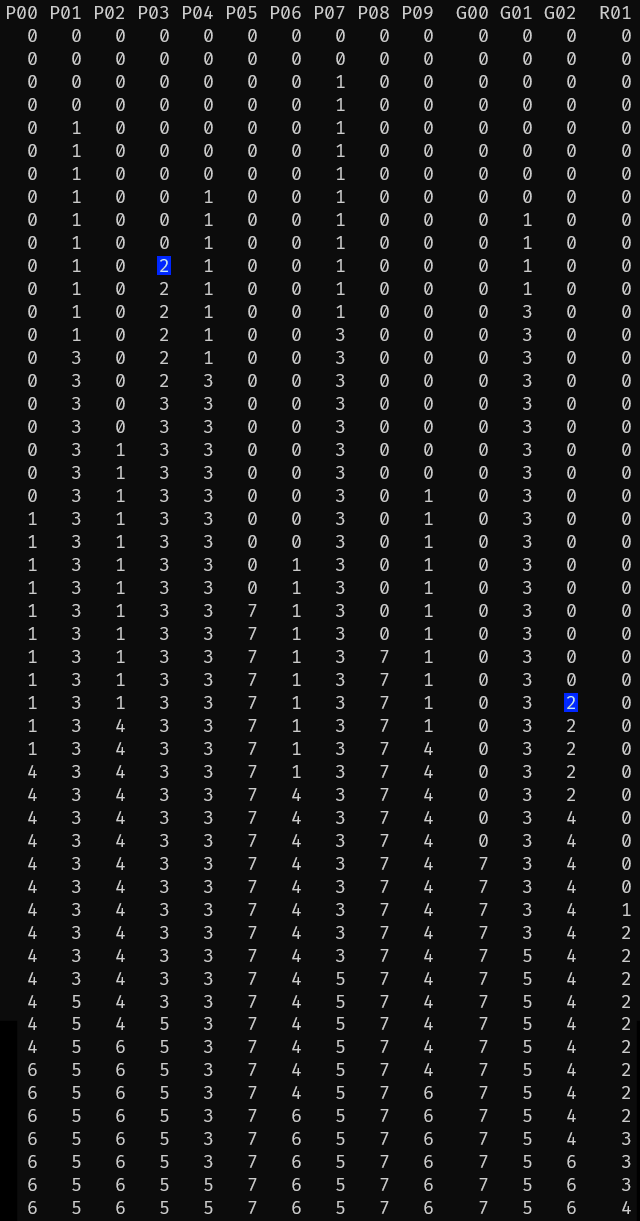
- Passado algum tempo, mudar para estado 4 (ENDING\_GAME) e terminar;

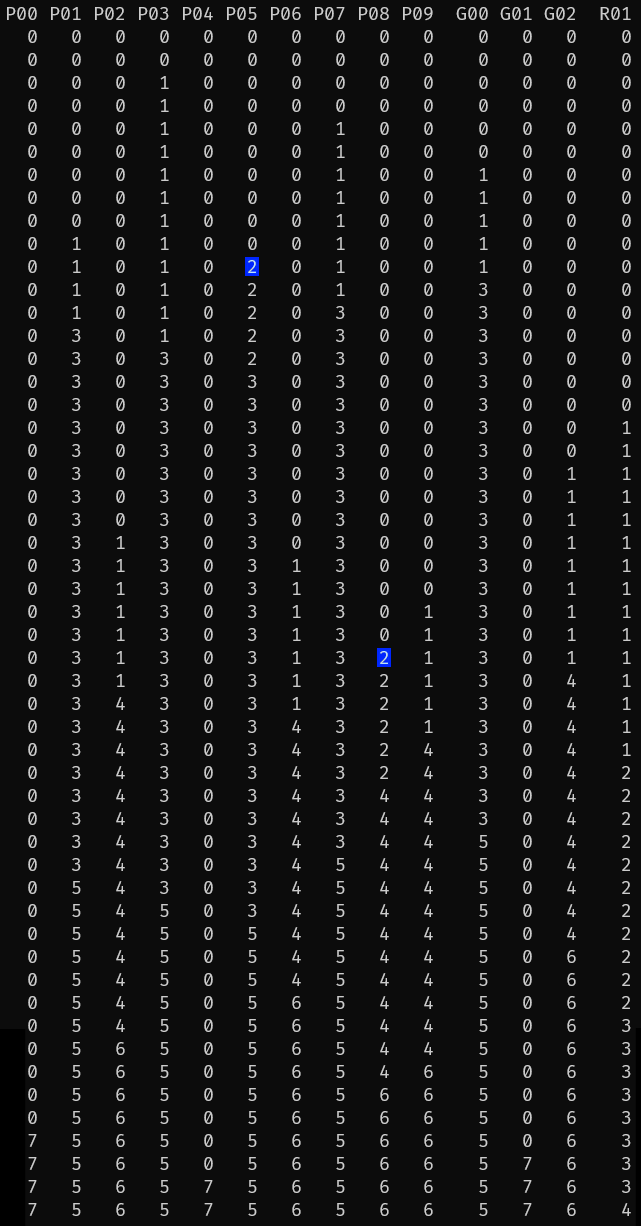
De maneira a escolhermos um exemplo de execução aleatório, corremos o programa 1000 vezes (com o script ./run) e escolhemos uma execução aleatória dentro das 10 últimas.

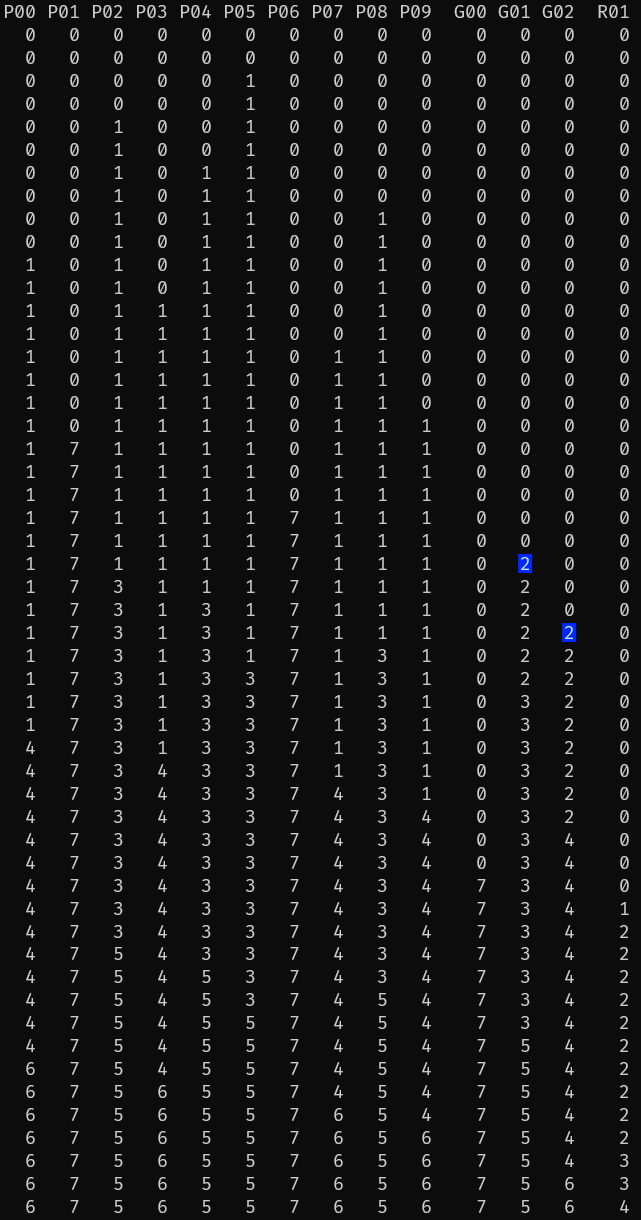
Analisando esta execução (Exemplo Player e Goalie a formar equipa) podemos ver que:

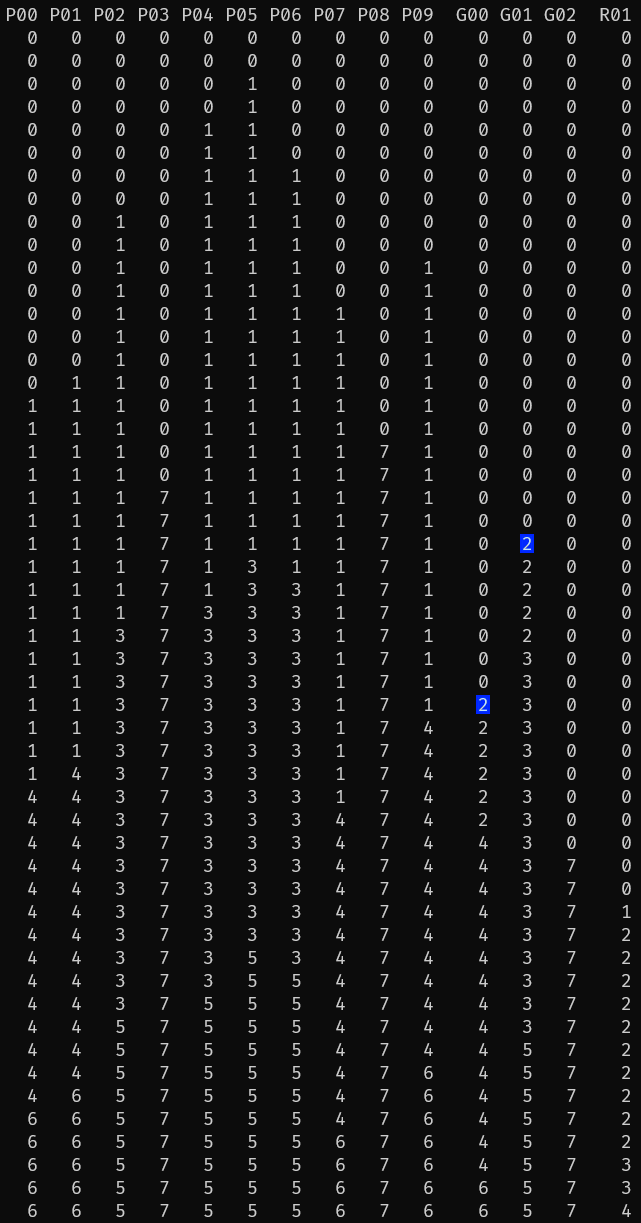
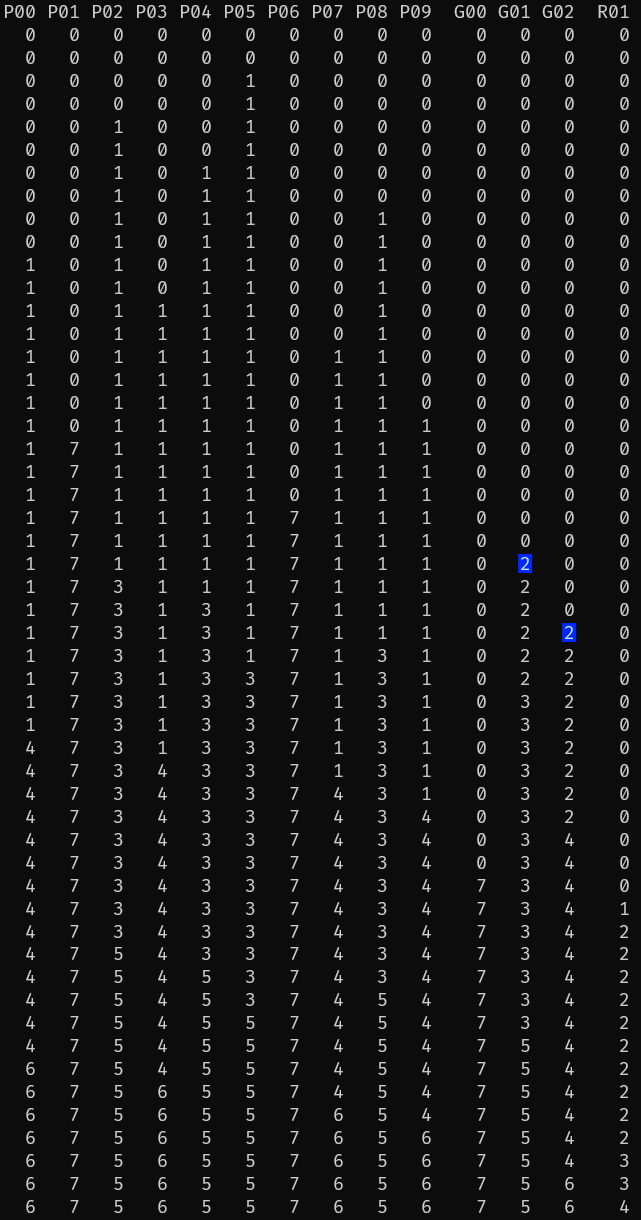
* Todos os players, goalies e o referee começaram com estado 0.
* O player P07 foi o primeiro a chegar, e como não havia jogadores suficientes para formar equipa mudou o estado para 1. O mesmo ocorreu com os players P01 e P04, segundo e terceiro a chegar respetivamente.
* O goalie G01 foi o primeiro goalie a chegar, mas ainda não havia players suficientes, logo alterou o seu estado para 1;
* O player P03 chegou e reparou que já havia jogadores suficientes para formar equipa então mudou o seu estado para 2;
* O goalie G01 e os players P07, P01 e P04 mudaram o seu estado para 3, ou seja, pertencem à equipa 1 e estão à espera do início do jogo. O player P03, que foi o jogador que formou a equipa, é o último a trocar de estado para 3.
* Chegam os players P02, P09, P00 e P06 e mudam o seu estado para 1, pois não existem jogadores suficientes para formar equipa.
* Chegam os players P05 e P07 e verificam que chegaram atrasados, pois já existem players suficientes para as duas equipas, então alteram o seu estado para 7.
* Chega o goalie G02, verifica que existem jogadores suficientes e forma equipa, mudando o seu estado para 2.
* Os players P02, P09, P00 e P06 alteram o seu estado para 4 pois pertencem à equipa 2 e estão à espera que o jogo comece. Após todos os membros da sua equipa terem trocado o seu estado, o jogador que formou a equipa G02 altera o seu estado para 4.
* Chega o goalie G00 e verifica que existem goalies suficientes para as duas equipas, então altera o seu estado para 7.
* O referee R01 chega e altera o seu estado para 1, pois está à espera das equipas. Verifica rapidamente que as equipas estão formadas e altera o seu estado para 2.
* Os jogadores começam a alterar os seus estados para 5 ou 6 consoante a sua equipa.
* O referee altera o seu estado para 3, assinalando o início da partida. Neste caso aconteceu uma situação em que o referee começou a partida antes do goalie G02 e do player P04 alterarem o seu estado para 5 ou 6. Isto é normal pois o goalie não verifica se todos os jogadores trocaram de estado para 5 ou 6 antes de trocar a equipa. Pode ocorrer ainda uma situação, pelo mesmo motivo, em que o referee acabe o jogo antes de alguns jogadores trocarem o seu estado para 5 ou 6. Para resolvermos ambas as situações seria necessário adicionar mais um semáforo de maneira a obrigar o referee a verificar se todos os jogadores mudaram o seu estado antes de iniciar o jogo.
* Passado algum tempo o referee muda o seu estado para 4, assinalando o fim do jogo. Todos os jogadores terminam.

Foram feitas e analisadas mais execuções além desta, todas com resultados semelhantes aos descritos acima. Sendo assim, podemos concluir que o programa se encontra a funcionar como esperado.

Exemplo Player e Goalie a Exemplo Player e Player a

formar equipa formar equipa

Exemplo Goalie e Goalie a formar Exemplo Goalie e Goalie a formar

equipa e no estado 2 ao mesmo tempo equipa com algum intervalo entre eles

Exemplo onde alteramos as constantes do ficheiro *probConst.h* relativas ao número de players total, que passou a 15, o número de goalies total, que passou a 5, o número de players em cada equipa, que passou a 6, e o número de goalies em cada equipa,que passou a 2.

Com este exemplo podemos verificar que o programa funciona para tamanhos variáveis de equipas e número total de jogadores.

