

A wooden checkers board with a light and dark brown checkered pattern. White pieces are on the left side, and black pieces are on the right side. The pieces are arranged in a standard starting position for a game of checkers.

Damas

Projeto Programação Declarativa

Projeto Final

Problemas encontrados:

- A implementação foi feita usando JPL.
- Erro ao tentar comer a peça do adversário.
- O jogo só permite jogar humano x humano.
- Da maneira que o jogo foi implementado o jogador não é obrigado a comer sempre que possível.

Erro ao tentar comer a peça do adversário

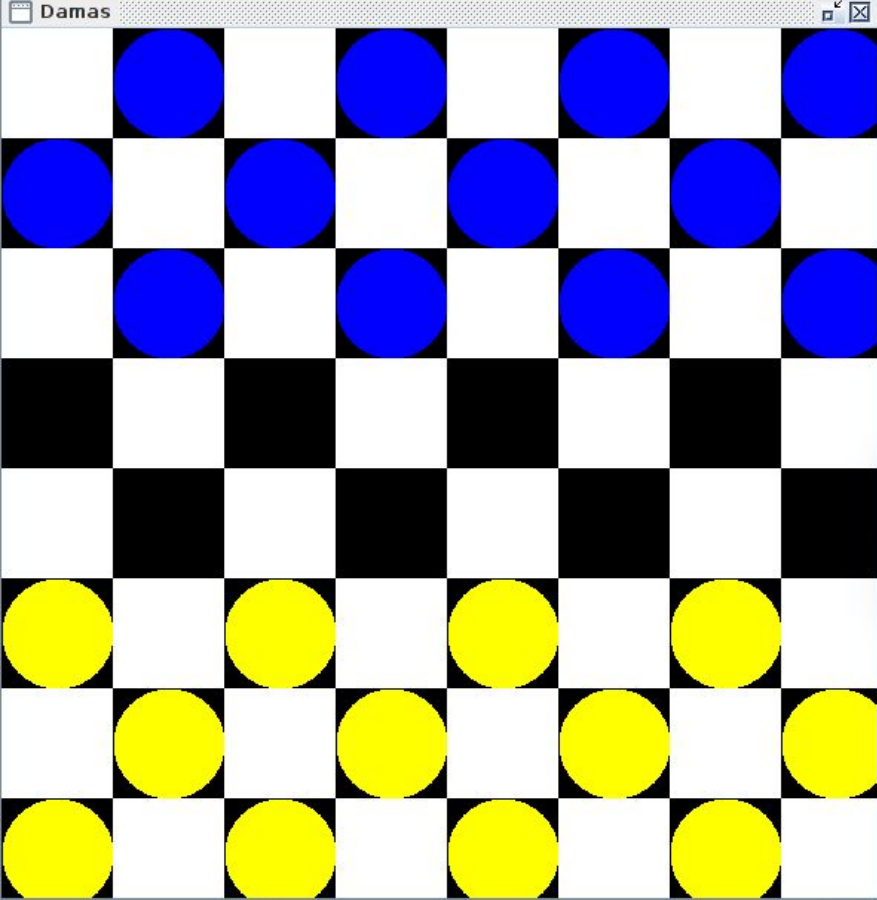
```
ERROR: Arguments are not sufficiently instantiated
ERROR: In:
ERROR:   [17] atomic_list_concat(_37684,',',_37688)
ERROR:   [16] interface([_37716,_37722|...]) at /home/jaine/Dropbox/Universidade/3periodo/Prog Declarativa/Projeto Final - Damas/Damas/damas/Projeto DAMAS/damasq.pl:262
ERROR:   [15] aux(y,[_37760,_37766|...],'Branco',_37754,44) at /home/jaine/Dropbox/Universidade/3periodo/Prog Declarativa/Projeto Final - Damas/Damas/damas/Projeto DAMAS/damasq.pl:324
ERROR:   [13] aux(y,[[44|...],...|...],'Branco',_37798,44) at /home/jaine/Dropbox/Universidade/3periodo/Prog Declarativa/Projeto Final - Damas/Damas/damas/Projeto DAMAS/damasq.pl:329
ERROR:   [11] processa([[26|...],...|...],'Branco',_37846) at /home/jaine/Dropbox/Universidade/3periodo/Prog Declarativa/Projeto Final - Damas/Damas/damas/Projeto DAMAS/damasq.pl:286
ERROR:   [10] processa([[26|...],...|...],'Preto',_37892) at /home/jaine/Dropbox/Universidade/3periodo/Prog Declarativa/Projeto Final - Damas/Damas/damas/Projeto DAMAS/damasq.pl:287
ERROR:    [9] processa([[1|...],...|...],'Branco',_37938) at /home/jaine/Dropbox/Universidade/3periodo/Prog Declarativa/Projeto Final - Damas/Damas/damas/Projeto DAMAS/damasq.pl:287
ERROR:    [7] <user>
ERROR:
ERROR: Note: some frames are missing due to last-call optimization.
ERROR: Re-run your program in debug mode (:- debug.) to get more detail.
Exception: (15) aux(y, _38006, 'Branco', _38010, 44) ?
```

Proposta para extensão


- Tentar executar com o JPL 7.4.
- Corrigir erro ao tentar comer a peça do adversário.
- Tornar comer sempre que possível obrigatório.
- Permitir jogar humano x computador.

1º passo: Tentar executar com o JPL 7.4.

- Foi adicionado ao projeto java a biblioteca jpl-7.4.0.jar.
- Programas em Prolog que usam java requer configurações adicionais para carregar bibliotecas Java.
- Usar LD_LIBRARY_PATH que teve que ser modificado para poder conter diretórios que contêm os arquivos libjava.so, libjni.so, libjsig.so.
- Buscar esses diretórios usando o seguinte comando: `dpkg --search libjsig.so libjava.so`
- `LD_LIBRARY_PATH=/usr/lib/jvm/java-8-openjdk-amd64/jre/lib/amd64/server:/usr/lib/jvm/java-8-openjdk-amd64/jre/lib/amd64/ swipl`




Entrada

 Digite a posição inicial

OK Cancelar



Progresso:

- Tentar executar com o JPL 7.4. 
- Corrigir erro ao tentar comer a peça do adversário.
- Permitir jogar humano x computador.
- Tornar comer sempre que possível obrigatório.

2º passo: Corrigir erro ao tentar comer a peça do adversário.

Com erro:

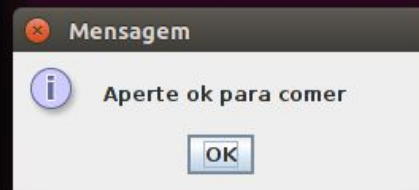
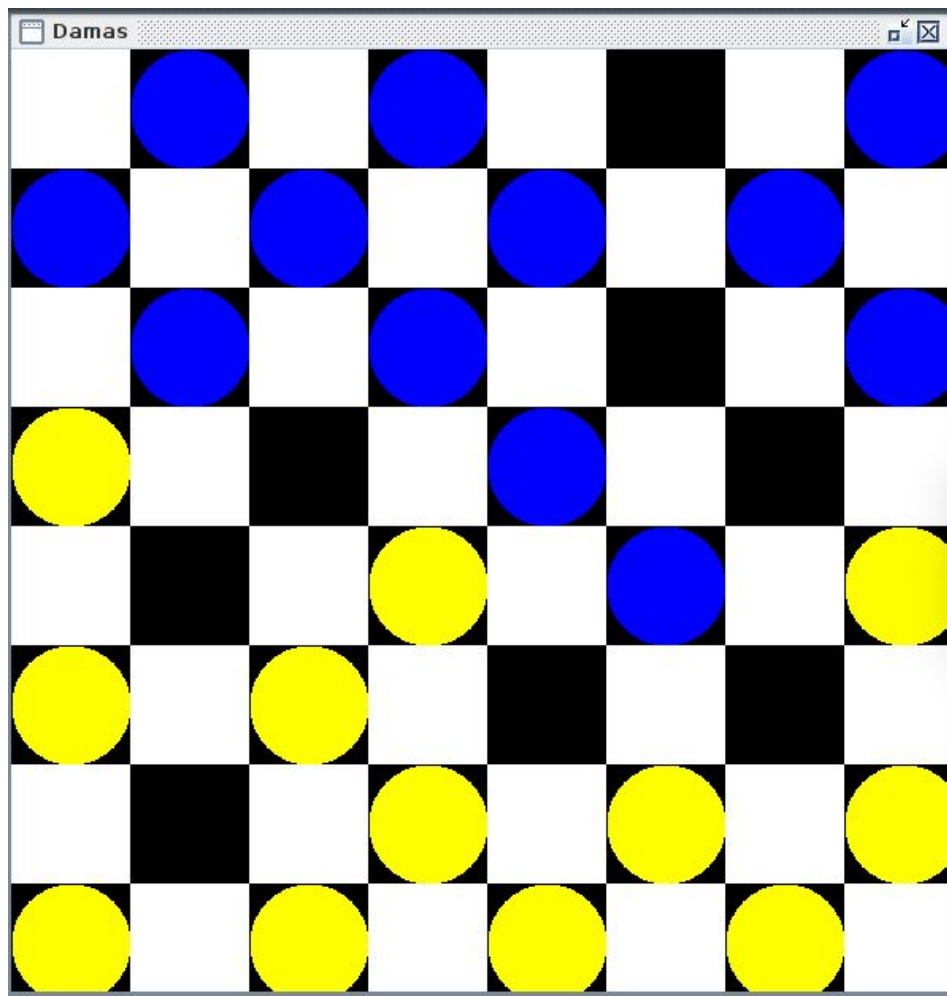
```
estado(1,Y,Y1,Turno,P):-lerC(C),aux(C,Y,Turno,Y1,P).
```

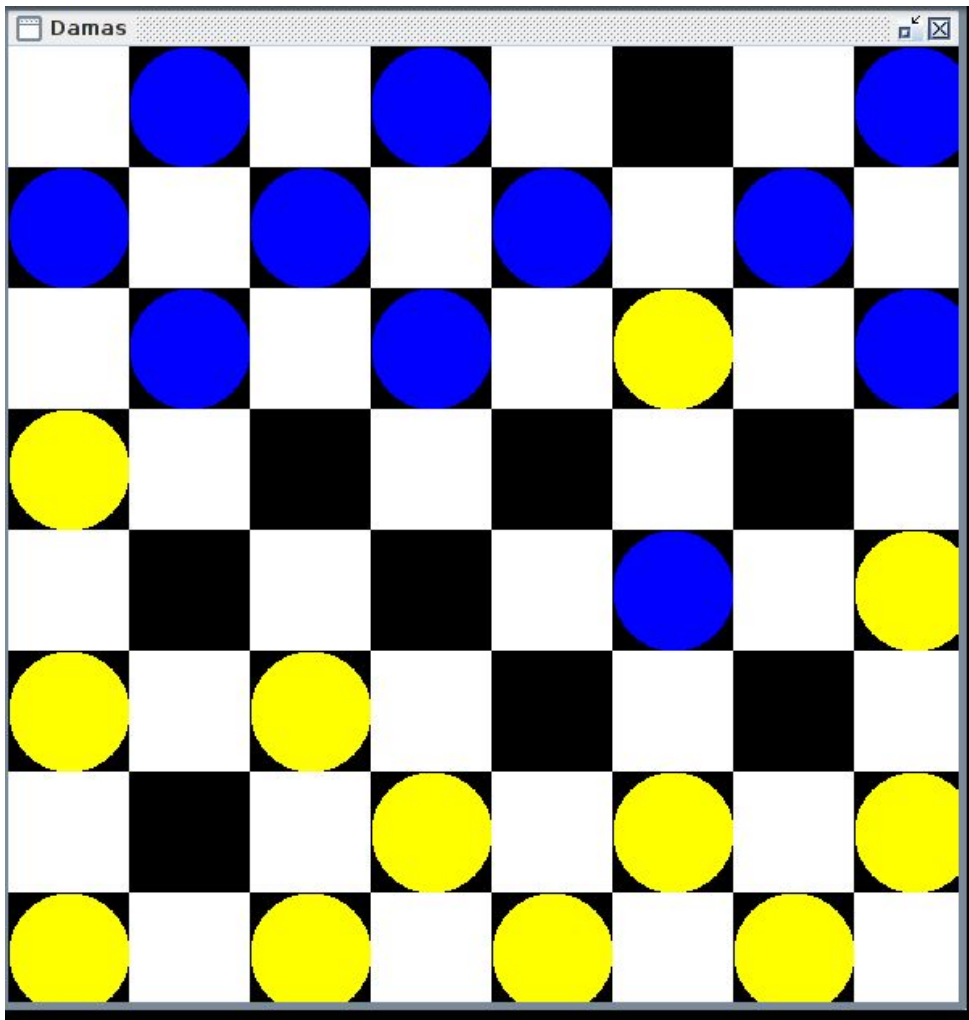
```
aux('y',[LB,LP,LD],Turno,Y,PI):-  
interface([LB,LP,LD]),  
lerPF(PF1),  
str_int(PF1,PF),  
ehDama(LD,PI,D),  
ehJogadaValida2(D,1,[LB,LP,LD],Turno,X,PI,  
PF),  
estado(1,X,Y,Turno,PF).
```

Sem erro:

```
estado(1,Y,Y1,Turno,P):-comerC(C),aux(C,Y,Turno,Y1,P).
```

```
aux('y',[LB,LP,LD],Turno,Y,PI):-interface([LB,LP,LD]).
```





*A regra deixada de fora foi: Se, no mesmo lance, se apresentar mais de um modo de tomar, é obrigatório executar o lance que tome o maior número de peças (lei da maioria). Pois, do jeito que foi implementado a extensão do jogo só é permitido capturar uma peça adversária por vez.

Progresso:

- Tentar executar com o JPL 7.4. ✓
- Corrigir erro ao tentar comer a peça do adversário. ✓
- Permitir jogar humano x computador.
- Tornar comer sempre que possível obrigatório.

3º passo: Permitir jogar humano x computador.



damas



damas.pl



tabuleiro.
jpg



Entrada



Pressione 1 para jogar contra a maquina e 2 para jogar contra outra pessoa

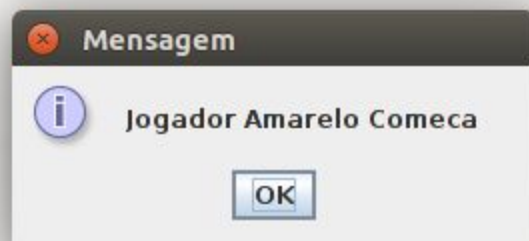
OK

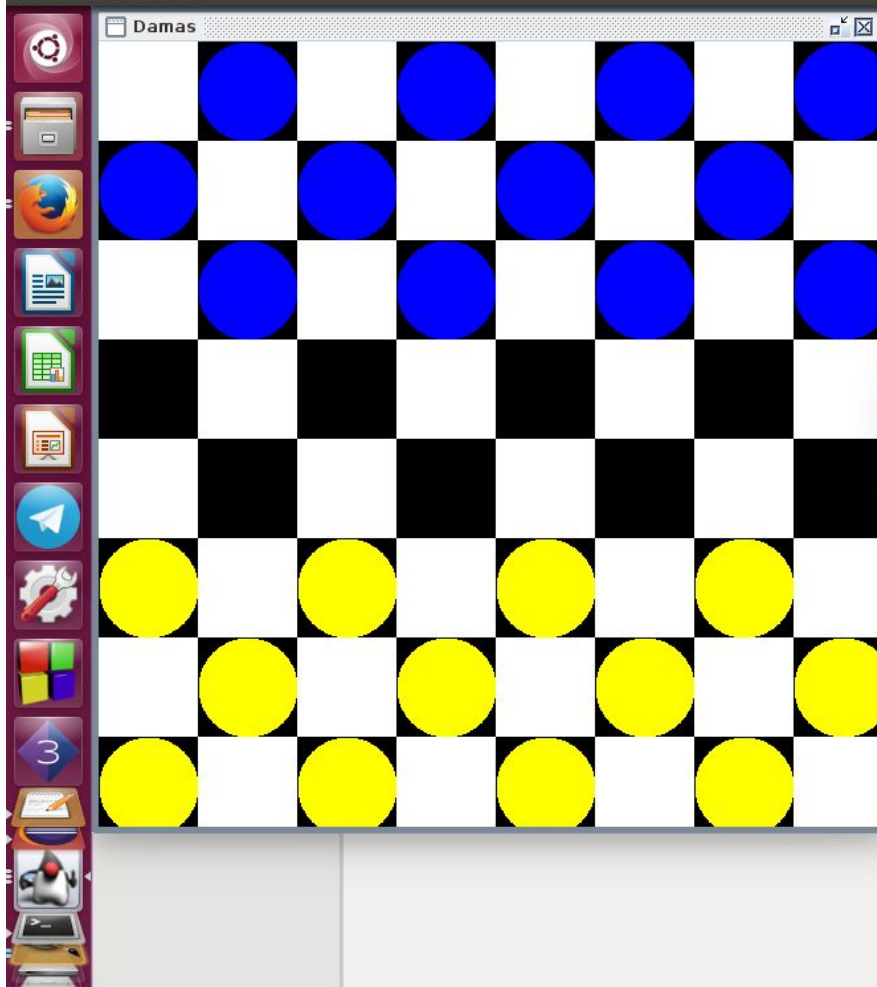
Cancelar



Tabuleiro de Damas

57	58	59	60	61	62	63	64
49	50	51	52	53	54	55	56
41	42	43	44	45	46	47	48
33	34	35	36	37	38	39	40
25	26	27	28	29	30	31	32
17	18	19	20	21	22	23	24
9	10	11	12	13	14	15	16
1	2	3	4	5	6	7	8





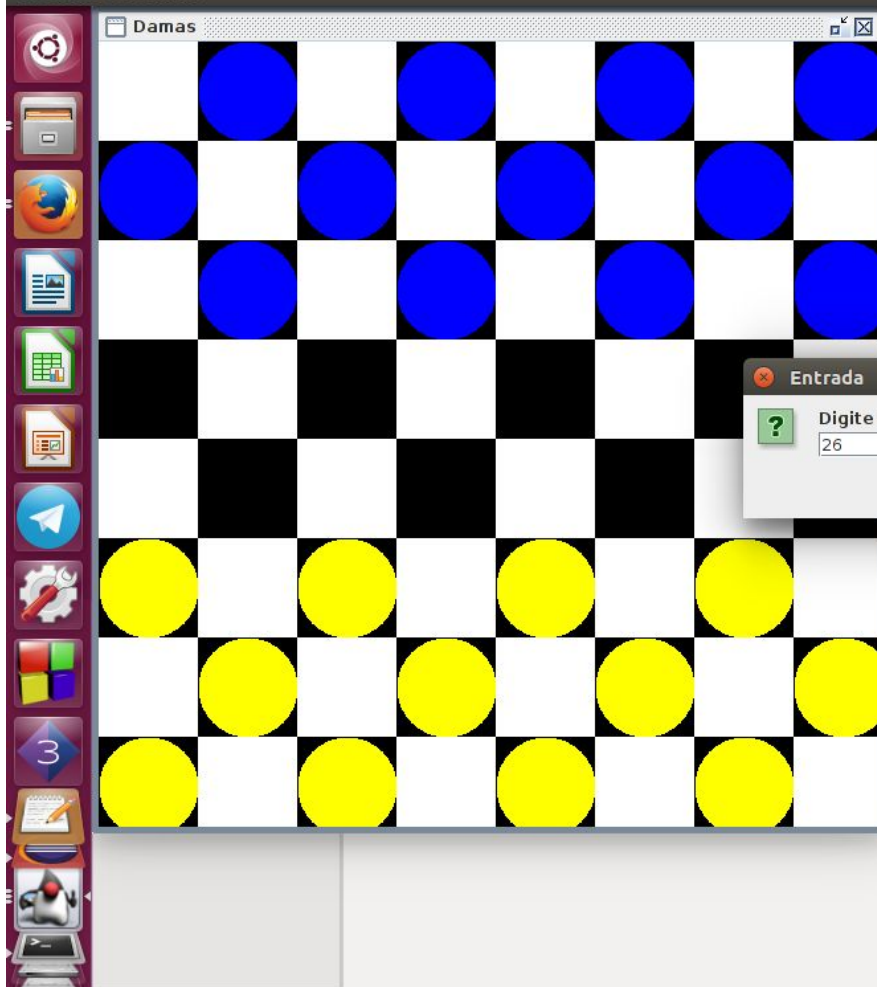
Entrada

? Digite a posicao inicial


17

OK Cancelar





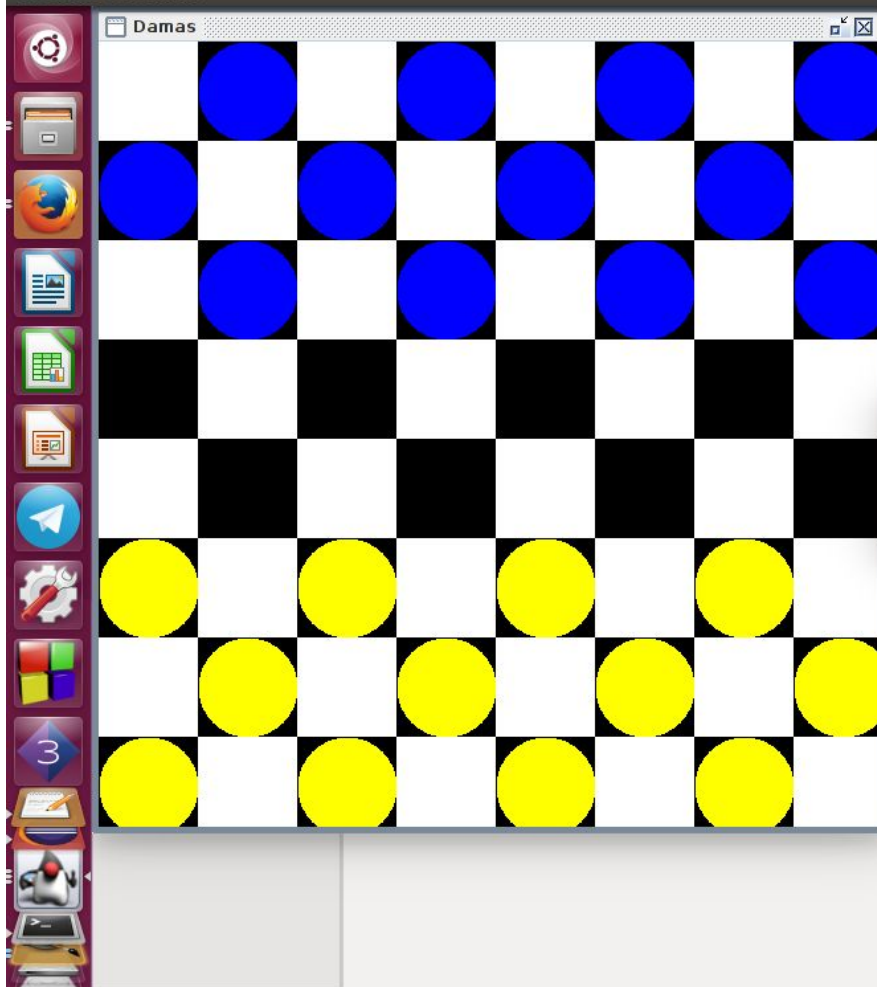
Entrada

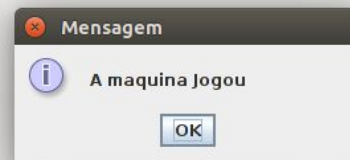
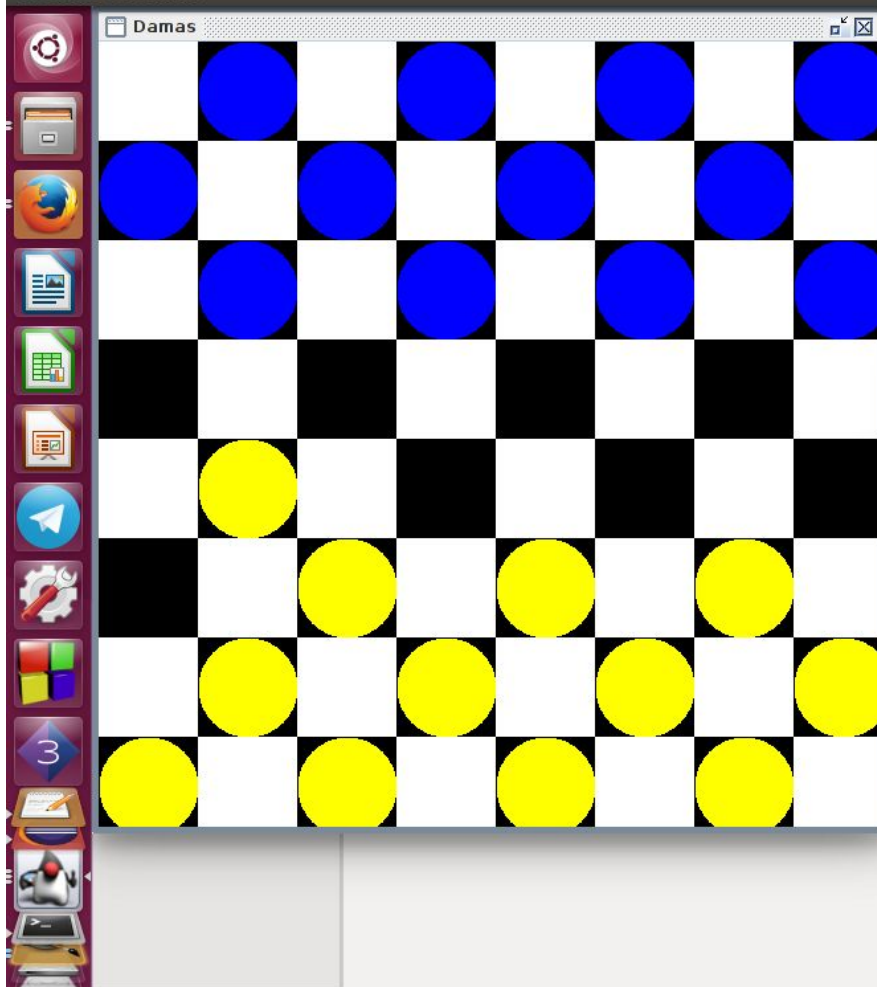

 Digite a posicao para a qual voce deseja mover a peca

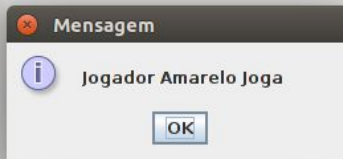
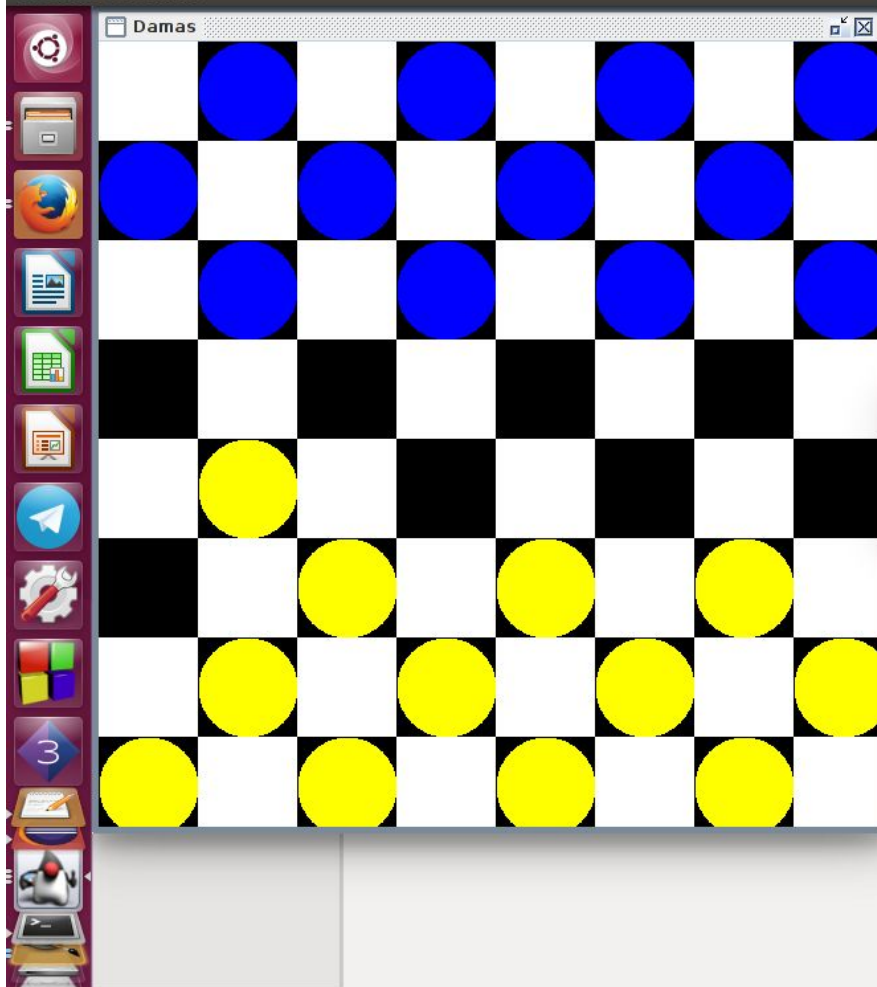
26

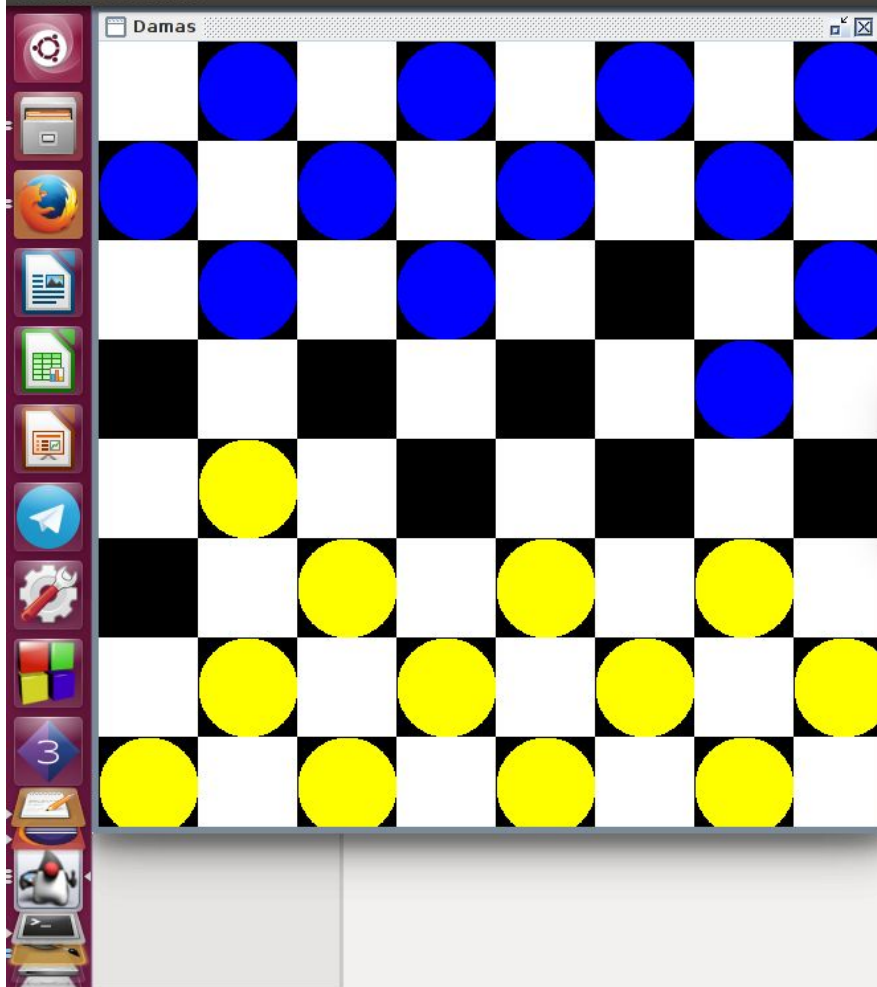
OK Cancelar









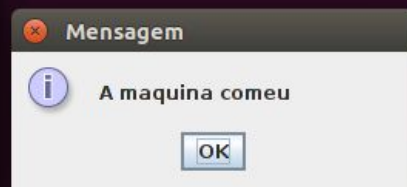
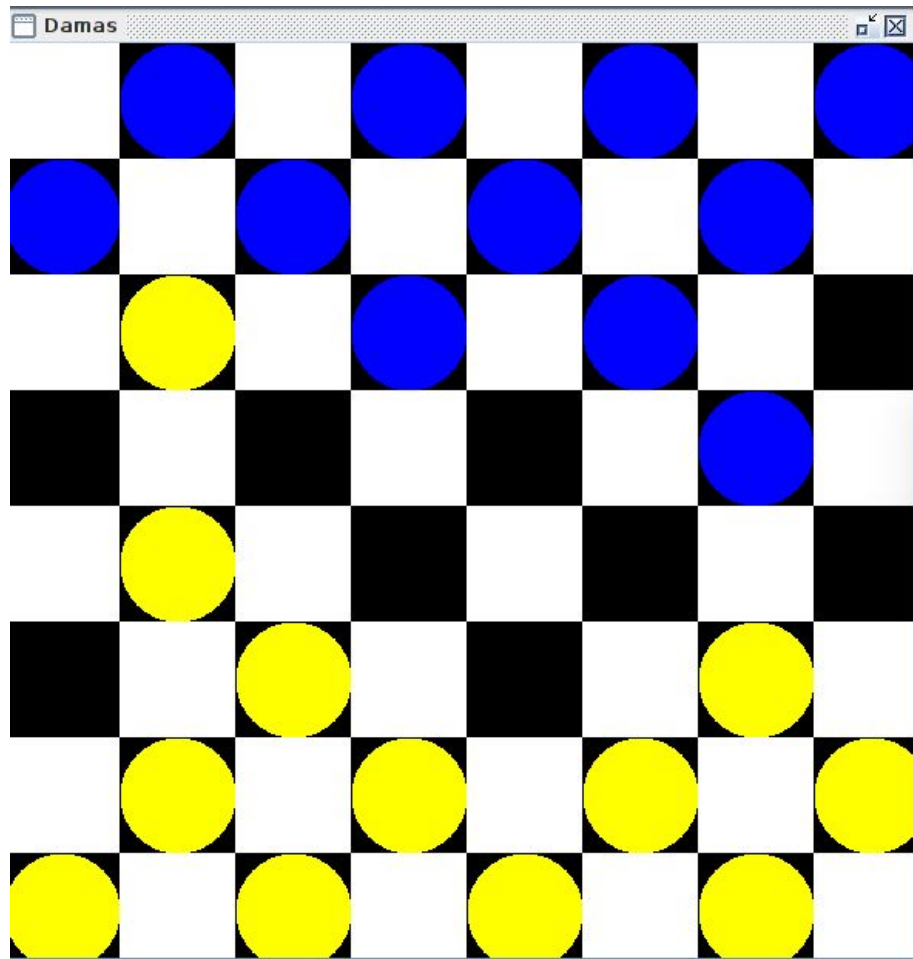


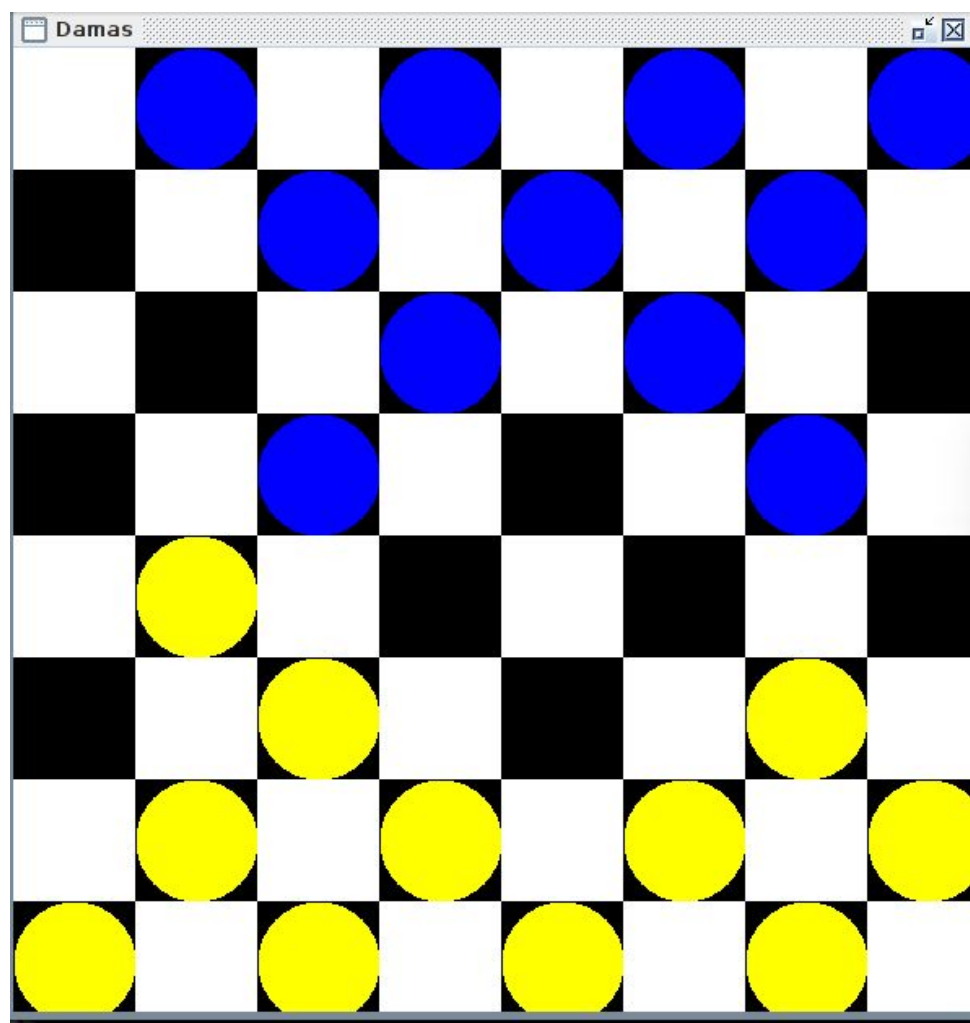
Entrada

? Digite a posicao inicial

OK Cancelar







Aspecto de agência

```
processa([LB,LP,LD], 'Preto',_, '1',0):-                %jogada da maquina
    interface([LB,LP,LD]),nl,
    pick_nums(PI1,LP,AUX,PF1),                          %lógica da maquina(aleatoria)
    ehDama(LD,PI1,D),
    ehJogadaValidaMaquina(D,E,[LB,LP,LD],Turno,X,PI1,PF1,C),
    estadoMaquina(E,X,Y,Turno,PF1),
    jpl_call('javax.swing.JOptionPane',showMessageDialog,[@null,'A maquina Jogou'],_),
    trocarTurno(Turno,Turno2),
    processa(Y,Turno2,_, '1',C).
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% CALCULO DA JOGADA DO COMPUTADOR %%%%%%%%%  
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
chute(N,S):-repeat, S is random(N). %jogada atual é aleatória
```

```
selecionaPF(1,PI1,PF1):- PF1 is PI1-7.
```

```
selecionaPF(2,PI1,PF1):- PF1 is PI1-9.
```

```
selecionaPF(3,PI1,PF1):- PF1 is PI1-18.
```

```
selecionaPF(4,PI1,PF1):- PF1 is PI1-14.
```

```
selecionaPF(5,PI1,PF1):- PF1 is PI1+7.
```

```
selecionaPF(6,PI1,PF1):- PF1 is PI1+9.
```

```
selecionaPF(7,PI1,PF1):- PF1 is PI1+18.
```

```
selecionaPF(8,PI1,PF1):- PF1 is PI1+14.
```

```
pick_nums(PI1,LP,AUX,PF1):-
```

```
    repeat, random_member(PI1,LP),chute(9,AUX),selecionaPF(AUX,PI1,PF1).
```

Progresso:

- Tentar executar com o JPL 7.4. ✓
- Corrigir erro ao tentar comer a peça do adversário. ✓
- Permitir jogar humano x computador. ✓
- Tornar comer sempre que possível obrigatório.

3º passo: Implementar captura obrigatória.

%se a ultima posicao do adversario -7 for uma peca do jogador tento comer com a peça PF1+7

checaPossibilidadeDeComer(1,PF1,LB,S,PI):- PI is PF1-7,write(PI),member(PI,LB),S is PF1+7.

%se a ultima posicao do adversario -9 for uma peca do jogador tento comer com a peça PF1+9

checaPossibilidadeDeComer(2,PF1,LB,S,PI):- PI is PF1-9,write(PI),member(PI,LB),S is PF1+9.

%se a ultima posicao do adversario +7 for uma peca do jogador tento comer com a peça PF1-7

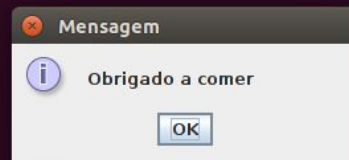
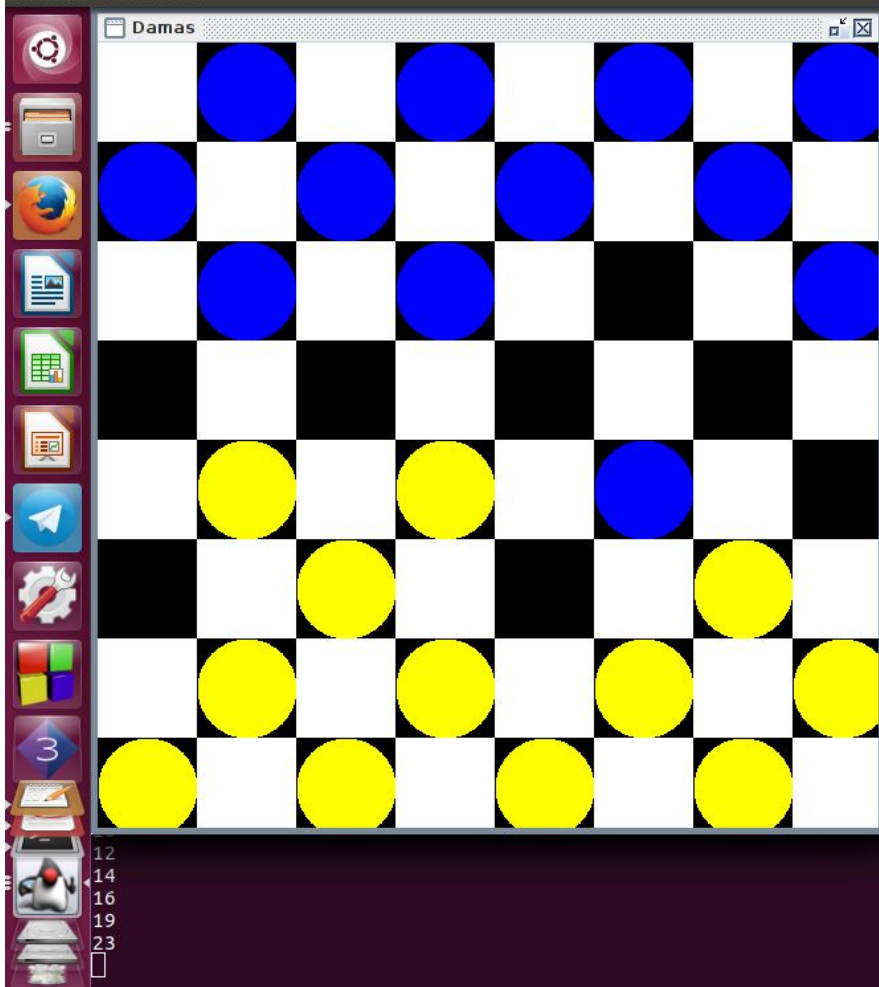
checaPossibilidadeDeComer(3,PF1,LB,S,PI):- PI is PF1+7,write(PI),member(PI,LB),S is PF1-7.

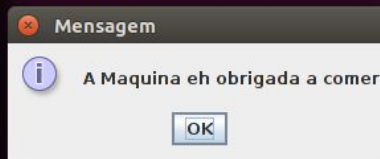
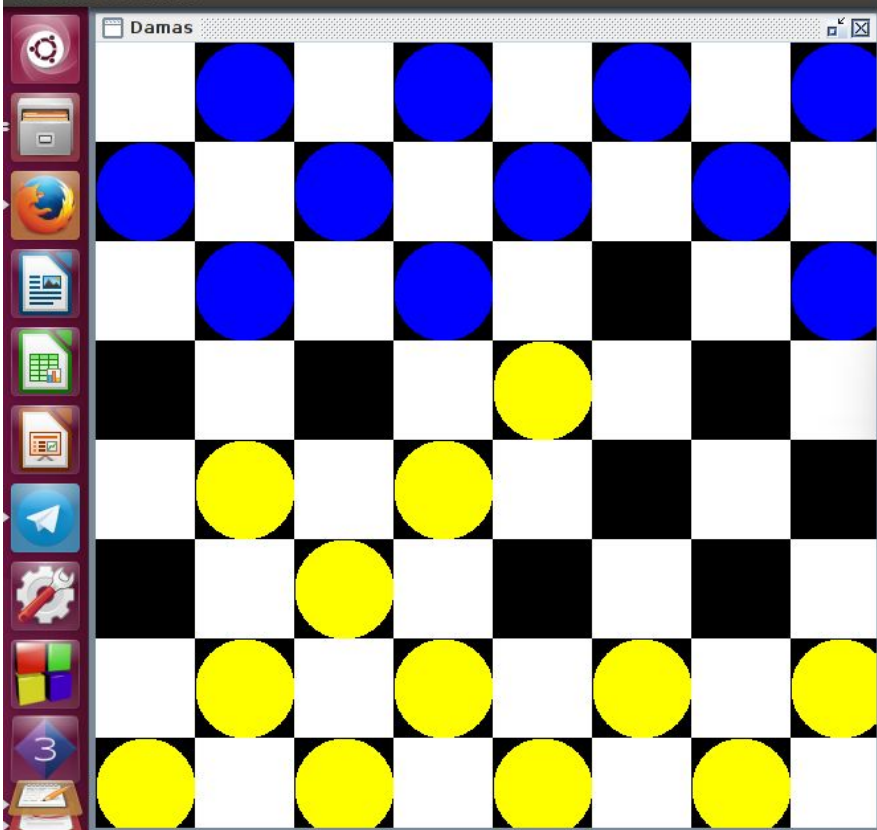
%se a ultima posicao do adversario +9 for uma peca do jogador tento comer com a peça PF1-9

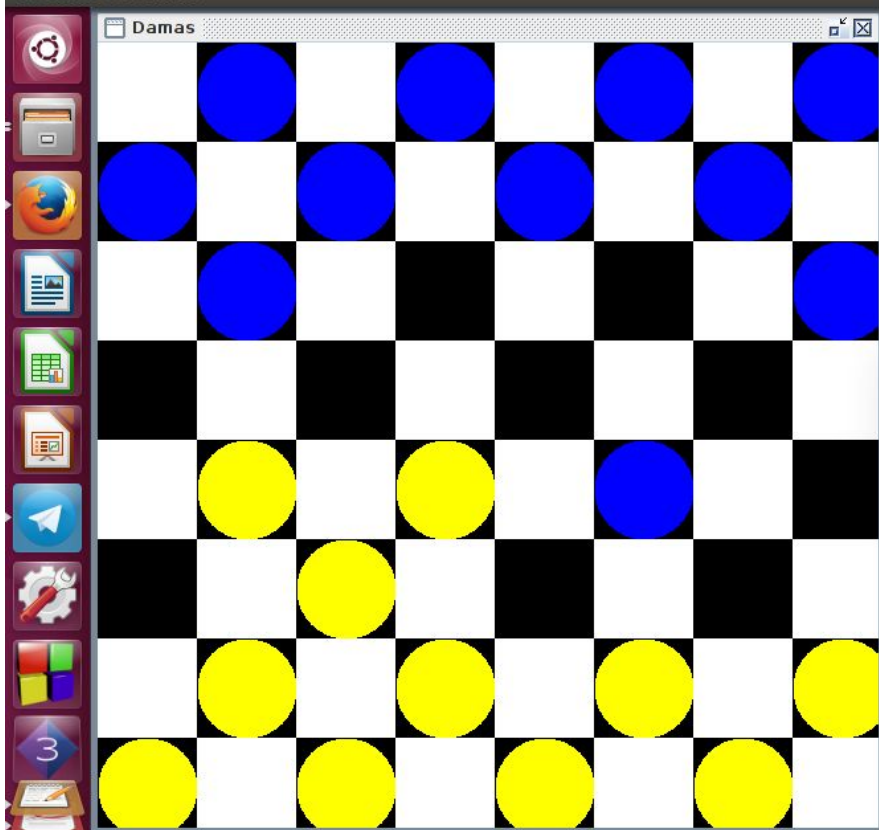
checaPossibilidadeDeComer(4,PF1,LB,S,PI):- PI is PF1+9,write(PI),member(PI,LB),S is PF1-9.

3º passo: Implementar captura obrigatória.

Vale ressaltar que a implementação da captura obrigatória foi implementada tanto para as jogadas dos humanos na opção humano x humano, como para as jogadas do humano e da máquina na opção humano x computador.







Entrada

? Digite a posicao inicial

OK Cancelar



12
14
16
19
23302844C LASTZA023213739

Progresso:

- Tentar executar com o JPL 7.4. ✓
- Corrigir erro ao tentar comer a peça do adversário. ✓
- Permitir jogar humano x computador. ✓
- Tornar comer sempre que possível obrigatório. ✓

OBRIGADO!