Lidia Sumie Yano

Mestranda em Fisiologia Geral do Instituto de Biociências da Universidade de São Paulo (IB-USP) no Laboratório de Fisiologia Respiratória e Metabolismo Energético.

Título do projeto de pesquisa: “Efeitos de altas temperaturas e do termoperíodo sobre a capacidade metabólica dos tecidos em Lithobates catesbeianus (Shaw, 1802).”

Meus Exercícios

Link para a página com os exercícios resolvidos → Exercícios

Proposta do Trabalho Final

Plano A (Principal): JO-KEN-PÔ!

Contextualização

O Jokenpô tradicional é uma modalidade de jogo infantil praticado entre duas pessoas que demanda decisões rápidas e auxilia no desenvolvimento cognitivo, mas isso não impede que adultos também o pratiquem pra decidir quem vai lavar a louça

O jogo é simples: Ambos os jogadores escolhem ao mesmo tempo uma opção de objeto (pedra, papel ou tesoura) e sinalizam um formato característico com a mão referente a cada objeto escolhido. O objeto vencedor é definido por uma regra cíclica de forma que cada um dos objetos tem as mesmas chances de “vencer” e “perder” para um determinado objeto, como esquematizado na imagem abaixo:

As únicas opções de final são: Jogador 1 ganha, Jogador 2 ganha ou empate. O empate ocorre quando o mesmo elemento é apresentado pelos dois jogadores, sendo assim, repete-se o procedimento até que um deles ganhe.

Atualmente este jogo apresenta muitas variantes dependendo do nível de complexidade e possibilidades, como por exemplo o aumento na número de elementos que, além de aumentar a complexidade das relações de “perder” e “ganhar” entre eles permite com que mais jogadores participem da mesma partida. Entretanto, para esta atividade proposta, seguirá a regra tradicional

jogando o operador contra a máquina e adicionado da possibilidade de escolha em ser realizado apenas uma vez, melhor de 3 ou melhor de 5 partidas. **Que vença o melhor!**

Planejamento da Função

1) Entrada

- `jokenpo(x)`
  - x = é o número de partidas sendo (1) para uma vez, (3) para melhor de 3 ou (5) para melhor de 5 partidas.

2) Verificação de Parâmetros

- x = Só pode ser 1, 3 e 5.
  - Se não, escreve: “Ow! Você só pode escolher entre 1, 3 ou 5 partidas!!”

3) Pseudo-código

1. Cria uma matriz `tab.res` de quem ganha, perde e empata.
2. Cria uma tabela de pontuação chamada `tab.pontos` com 1 linha e 2 colunas com valores NAs e títulos das colunas como “operador” e “computador”:
3. Escreve a instrução geral do jogo: **“Escolha entre pedra, papel ou tesoura e o programa escolherá um destes elementos aleatoriamente. Saiba que tesoura ganha de papel, papel ganha de pedra e pedra ganha de tesoura. Boa sorte!”**
4. Se `jokenpo(1)`:
    1. Comando interativo: “Escolha uma opção: (pedra/papel/tesoura)” e guarda no objeto `linha1`
    2. Escolhe aleatoriamente um das opções e guarda no objeto `coluna1`
    3. Busca `[linha1,coluna1]` da matriz `tab.res`
        1. Se o resultado for `empate`:
            1. repete a interação, a escolha aleatória, e a busca do resultado até que o resultado seja diferente de `empate`
        2. Se o resultado for `ganha` :
            1. Adiciona “1” na coluna [1] do objeto `tab.pontos` e adiciona “0” na coluna [2]
            2. Retorna a tabela `tab.pontos` e uma frase: **“Parabéns, você é o vencedor!”**
        3. Se o resultado for `perde` :
            1. Adiciona “0” na coluna [1] de `tab.pontos` e adiciona “1” na coluna [2]
            2. Retorna a tabela `tab.pontos` e a frase: **“Que pena… mais sorte na próxima vez!”**
5. Se `jokenpo(3)`:
    1. Gera um ciclo:
        1. Comando interativo: “Escolha uma opção: (pedra/papel/tesoura)” e guarda no

objeto `linha(i)`
2. Escolhe aleatoriamente um das opções e guarda no objeto `coluna(i)`
3. Busca `[linha(i),coluna(i)]` da matriz `tab.res`
    1. Se o resultado for `empate`:
        1. não faz nada e retorna as opções, uma frase de empate
    2. Se o resultado for `ganha`:
        1. Soma 1 a coluna [1] da `tab.pontos` , retorna `tab.pontos`, e as opções e uma frase **“Parabéns, você é o vencedor!”**
    3. Se o resultado for `perde`:
        1. Soma 1 a coluna [2] da `tab.pontos`, retorna `tab.pontos`, as opções e uma frase **“Que pena... mais sorte na próxima vez!”**
    4. Se o `OPERADOR == 2`: para e retorna `tab.pontos` e a frase **“Fim de jogo! Voce ganhou!”**
    5. Se o `COMPUTADOR == 2`: para e retorna `tab.pontos` e a frase **“Fim de jogo! Não foi dessa vez...tente da próxima!”**
    6. Se o `OPERADOR == COMPUTADOR`: retorna a frase: **“Empate de pontos! Este será o último jogo!”**

6. Se `jokenpo(5)`:
    1. Gera um ciclo:
        1. Comando interativo: “Escolha uma opção: (pedra/papel/tesoura)” e guarda no objeto `linha(i)`
        2. Escolhe aleatoriamente um das opções e guarda no objeto `coluna(i)`
        3. Busca `[linha(i),coluna(i)]` da matriz `tab.res`
            1. Se o resultado for `empate`:
                1. não faz nada e retorna as opções, uma frase de empate
            2. Se o resultado for `ganha`:
                1. Soma 1 a coluna [1] da `tab.pontos` , retorna `tab.pontos`, e as opções e uma frase
            3. Se o resultado for `perde`:
                1. Soma 1 a coluna [2] da `tab.pontos`, retorna `tab.pontos`, as opções e uma frase de perdeu
            4. Se o `OPERADOR == 3`: para e retorna `tab.pontos` e a frase **“Parabéns, você é o vencedor!”**
            5. Se o `COMPUTADOR == 3`: para e retorna `tab.pontos` e a frase **“Fim de jogo! Não foi dessa vez...tente da próxima!”**
            6. Se o `OPERADOR == 2` e `COMPUTADOR == 2`: retorna a frase: **“Empate de pontos! Este será o último jogo!”**

Julia Barreto

Oi, Lidia, tudo bele? Achei a função divertida, um pouco simples mas acho que você pode pensar em maneiras de como complementá-la para ficar mais interessante pro usuário e desafiadora para ti. Podemos seguir com ela se a gente tentar pensar em como aprimorar. Por exemplo, fiquei me perguntando o porquê de limitar o número de partidas a até 5... Porque não usar um número real positivo como argumento e seguir o jogo? Se por acaso fosse par ou desse empate, valeria oferecer a rodada de desempate.

Use sua criatividade para caprichar mais, sei que Jokenpo é um jogo simples por essência mas lembre-se que você tem liberdade poética para produzir uma novidade em prol do seu aprendizado! Me escreve se quiser conversar sobre ideias e/ou qualquer dúvida que venha a ter. Estou à disposição Julia Barreto

Plano B: TOP SECRET!

Ao mesmo tempo que a internet tem facilitado nossas vidas, o acesso a informações pessoais tem se tornado mais acessível para qualquer outra pessoa. Redes sociais, bancos etc, fornecem uma senha pessoal ou pedem para que o usuário as criem para facilitar a memorização, mas nem sempre estes códigos pessoais de acesso apresentam um nível de dificuldade suficiente para impedir de serem hackeadas. Sendo assim, é comum que as empresas exijam uma série de características durante a criação da senha para evitar este tipo de complicação como, por exemplo, incluir letras maiúsculas e minúsculas, números e até caracteres especiais. Por outro lado isto demanda tempo e, convenhamos, criatividade também.

Portanto, o objetivo desta função é oferecer senhas (podendo ser aleatórias ou não) baseada nem alguns requisitos fornecidos pelo usuário.

1) Entrada

- senha(`n.letras`, `n.LETRAS`, `n.num`, `n.caract` , `palavra`, `mix`, `invertido`)
- `n.letras` = número de letras do alfabeto minúsculas (classe:interger, `n.letras`>= 0)
- `n.LETRAS` = número de letras do alfabeto maiúsculas (classe:interger, `n.LETRAS`>= 0)
- `n.num` = quantidade de números (classe:interger, `n.num`>= 0)
- `n.caract` = número de caracteres especiais (classe:interger, `n.caract`>= 0)
- `palavra` = vetor contendo as letras da palavra separadas dentro do objeto
- `mix` =
  - Se TRUE , realizará mistura aleatória de `n.letras` e/ou `n.LETRAS` e/ou `n.num` e/ou `n.caract` e/ou `palavra` .
  - Se FALSE, é mantido nesta mesma ordem.
- `invertido` =
  - Se TRUE, realizará a substituição das letras de `palavra` por números que possuem aparência semelhante.
  - Se FALSE, é mantido.

2) Verificação de Parâmetros

- `n.letras`, `n.LETRAS`, `n.num`, `n.caract` = são números inteiros, maiores ou iguais a zero. Se não, escreve: **“ O número precisa ser inteiro e maior ou igual a zero”**
- `palavra` = é um vetor que contêm apenas uma letra em cada casa. Se não, escreve: **“A palavra deve ser escrita dentro de um vetor cada letra individualmente (ex. 'p','a','l','a','v','r','a')**

3) Pseudo-código

1. Criar um objeto com todas as letras minúsculas do alfabeto grego (`l`)
2. Criar um objeto com todas as letras maiúsculas do alfabeto grego (`L`)

3. Criar um objeto com os números de 0 a 9 (n)
4. Criar um objeto com os caracteres especiais básicos mais comuns como #,@,% etc (ce)
5. Criar uma matriz com os números que se parecem com as letras maiúsculas e minúsculas (inv)(por exemplo: A é 4, S é 5 etc..)
6. Cria o objeto senha para armazenar as informações
7. Se `n.letras != 0` :
    1. Cria um objeto (`letras`) a partir do objeto `l` aleatoriamente com essa quantidade
    2. Adiciona ao objeto `senha`
8. Se `n.LETRAS != 0` :
    1. cria um objeto (LETRAS) a partir do objeto L aleatoriamente com essa quantidade
    2. Adiciona ao objeto `senha`
9. Se `n.num != 0` :
    1. cria um objeto (`num`) a partir do objeto n aleatoriamente com essa quantidade
    2. Adiciona ao objeto `senha`
10. Se `n.caract != 0` :
    1. cria um objeto (`caract`) a partir do objeto ce aleatoriamente com essa quantidade
    2. Adiciona ao objeto `senha`
11. Se `palavra != NULL` :
    1. Adiciona ao objeto `senha`
12. Se `mix == TRUE`
    1. Então pega o objeto `senha` e mistura aleatoriamente
13. Se `invertido == TRUE`
    1. Então substitui cada letra do vetor de `palavra` por sua correspondente da tabela `inv`
    2. Adiciona ao objeto `senha`

Saída

Retorna o objeto `senha`

> Julia Barreto
> Oi de novo! Achei a proposta interessante e útil mas, ainda sim, é simples. Seguir com essa seria necessário pensar em como poderia complementá-la para ficar mais instigante. Valeria estabelecer valores default para cada argumento e quem sabe tentar encaixar controles de fluxo. Sobre a saída, além da senha, poderia entregar uma dica ou lembrete para ela ou mesmo retomar quais são os requisitos que ela preencheu. Julia Barreto

---

O Trabalho Final

A proposta escolhida foi o Plano A: Jokenpo, porém com algumas modificações e acréscimos:

1. Segui a sugestão feita em colocar o primeiro argumento de número de partidas (`n.partidas`) como qualquer número inteiro, positivo e diferente de zero.
2. Adicionalmente, foi acrescentado um argumento que permite a escolha do `estilo` de jogo: tradicional ou mais complexa, com 5 elementos.
3. Foi acrescentado imagens com frases que aparecem em determinados momentos utilizando o pacote `magick`

4. Foi acrescentado também o pacote `crayon` para colorir frases

- Link para a página da função `jokenpo()` → jokenpo()
- Link para a página do `help` da função → help