

Versão mais detalhada das funções

Plano A: Preenchendo a ficha de "Dungeons and Dragons"

Entrada: `fichaDnD (atrib, ord, alinh, rac, clas, antec)`

`atrib` = Escolha do modo como os atributos serão escolhidos (vetor composto por caracteres escolhendo um destes, "aleatório" ou "fixo")

`ord` = Ordem dos atributos, do que terá o maior valor para o menor em sequência (vetor composto por 6 caracteres, "For", "Des", "Com", "Int", "Sab", "Car", na ordem desejada pelo usuário)

`alinh` = Escolha o seu alinhamento (vetor composto por caracteres escolhendo um destes, "LB", "LN", "LM", "CB", "CN", "CM", "NB", "N" ou "NM")

`rac` = Escolha da raça do personagem (vetor composto por caracteres escolhendo um destes, "anão", "elfo", "halfling", "humano", "draconato", "gnomo", "meio-elfo", "meio-orc" ou "tiefling")

`clas` = Escolha da classe do personagem (vetor composto por caracteres escolhendo um destes, "bárbaro", "bardo", "clérigo", "druída", "guerreiro", "monge", "paladino", "arqueiro", "ladino", "feiticeiro", "bruxo", "mago")

`antec` = Escolha dos antecedentes do personagem (vetor composto por caracteres escolhendo um destes, "acólito", "artesão de guilda", "artista", "charlatão", "criminoso", "eremita", "forasteiro", "herói do povo", "marinheiro", "nobre", "órfão", "sábio", "soldado")

Pseudo-código:

1- Criar os objetos que vão compor a saída:

Data frame *atributos*

Data frame *dados.gerais*

Data frame *teste.resist*

Data frame *habilidades.pro*

Data frame *habilidades.tot*

Vetor *magias*

Data frame *especiais*

Data frame *caracteristicas*

Data frame *equipamentos*

Data frame *proficiencias*

Data frame *info*

2- A partir do método para determinar os atributos que foi escolhido `atrib` fazer:

- Sorteio dos valores (uso de um **for** para colocar cada valor sorteado em um atributo diferente)
OU

- Usar os valores fixos determinados no livro do jogador

3- Com os valores definidos, ordená-los conforme o argumento `ord` e guardar no objeto *atributos*; criar vetor concatenando os dados de percepção passiva e iniciativa (vão compor futuramente o objeto *dados.gerais*)

4- Usar de **ifs** encadeados para cada possível raça `rac`. Para cada uma delas somar os valores necessários aos *atributos*; criar um vetor com os dados da velocidade (vai entrar futuramente em *dados.gerais*) e preencher o data frame *info*. Se as características alterarem outros objetos, criar os vetores ou fazer as alterações correspondentes

5- Usar de **ifs** encadeados para cada possível classe `class`. Preencher tanto o cabeçalho como parte dos dados de *especiais*; criar vetor contendo o hit dice, número de hit dices e o bônus de proficiência (vão entrar em *dados.gerais*); preencher o data frame *proficiencias*; preencher o data frame *equipamentos*; preenche parte do data frame *teste.resist*; preencher tanto o cabeçalho como os dados para o vetor *magias*; preencher o data frame *habilidades.pro*.

6- Usar de **ifs** encadeados para cada possível alinhamento `alinh`. Preencher o primeiro campo de *caracteristicas*.

7- Usar de **ifs** encadeados para cada possível antecedente `antec`. Preencher o resto do data frame *caracteristicas*. Para isso é necessário sortear as características dentre as diferentes possibilidades apresentadas no livro do jogador (usando **for**) e checar a compatibilidade com o alinhamento usando de **ifs**.

8.1- Acrescentar ao data frame *atributos* os modificadores dos atributos

8.2- Distribuir os valores dos testes de resistência no data frame *teste.resist* a partir dos modificadores dos atributos guardados em *atributos* e somar o bônus de proficiência quando for proficiente naquele teste de resistência

8.3- Distribuir os valores das habilidades no data frame *habilidades.tot* a partir dos modificadores dos atributos guardados em *atributos*

8.4- Cálculo dos bônus de ataque corpo a corpo e à distância a partir da proficiência (dentro de *dados.gerais*) + modificador de atributo (dentro de *atributos*) e acrescentar no data frame *especiais*

9- Retornar ao usuário texto informando sobre alguns cálculos que não serão feitos (como calcular a classe de armadura e como ela é afetada pelo uso de escudo E explicar como são alterados os valores das habilidades quando se é proficiente nelas) usando a função **message**

Retornar lista com todos os objetos criados

Plano B: Escolha de disciplinas da pós graduação

Entrada: `disciplinas.oferecidas (dia, inic, fim, cred, loc, per)`

`dia` = dia da semana a ser escolhido (vetor composto por caracteres, escolhendo um destes, "segunda", "terça", "quarta", "quinta", "sexta", "vários" e "qualquer")

`inic` = mês de início da disciplina (vetor composto por números inteiros onde $0 < inic < 13$)

`fim` = mês de término da disciplina (vetor composto por números inteiros onde $0 < fim < 13$)

`cred` = número de créditos oferecidos (vetor composto por números inteiros onde $0 < cred < 11$)

`loc` = Instituto ou faculdade em que serão procuradas as disciplinas (vetor composto por caracteres, escolhendo entre o nome de qualquer uma das Comissões de Pós Graduação listadas dentre as disciplinas oferecidas no Janus ou o argumento "todas")

`per` = Período do dia em que a disciplina será ministrada (vetor composto por caracteres, escolhendo um destes, "manhã", "tarde", "noite", "integral" e "qualquer")

Pseudo-código:

1- Criar o objeto que vai compor a saída: Data frame *disciplinas*

2- Ler as informações da página “Disciplinas oferecidas” do sistema Janus a partir do URL e criar um data frame contendo os códigos em uma coluna e os nomes das CPGs em outra

3- De acordo com o argumento `loc`, obter o código correspondente ao `loc` no data frame (usando indexação) e acessar a página do local escolhido pelo código obtido (usando do URL)

- Ler as informações da página do local escolhido e criar um data frame contendo os códigos em uma coluna e os nomes das Área de Concentração em outra

- Acessar as áreas de concentração usando os códigos no URL

- Ler as informações da página contendo as disciplinas e criar um data frame contendo os códigos em uma coluna e os nomes das disciplinas em outra

- Usando o código de cada disciplina, acessar pelo URL a página da Turma

- Fazendo buscas por indexação na página da Turma, guardar os dados referentes ao código da disciplina, nome da disciplina, data inicial, data final, número de créditos, dia da semana e horário no objeto *disciplinas*

4- De acordo com os argumentos de `dia`, `inic`, `fim`, `cred` e `per`, fazer buscas em *disciplinas* usando de indexação para selecionar apenas as disciplinas compatíveis com as exigências dos argumentos

- Caso existam disciplinas que correspondam a todos os argumentos pedidos retornar o data frame *disciplinas* com a lista das disciplinas selecionadas e as informações sobre as mesmas

- Caso não existam disciplinas que correspondam a todos os argumentos pedidos retornar “Infelizmente não foram encontradas disciplinas com as características desejadas”

Saída: Data frame *disciplinas* com as informações das disciplinas filtradas pela função (n linhas - número de disciplinas, vai variar de acordo com a busca- e 7 colunas)

From:

<http://ecor.ib.usp.br/> - ecoR

Permanent link:

http://ecor.ib.usp.br/doku.php?id=05_curso_antigo:r2018:alunos:trabalho_final:arthur.cambler:funcoes_detalhadas

Last update: 2020/08/12 06:04