Diagnosticador IST

Release v1.0

Gustavo Silva Guimarães e João Geraldo Borges Sales

23 jul. 2021

Contents:

CAPÍTULO 1

diagnosticadorISTsPython

1.1 Módulo Main

Nessa parte do sistema concentram-se funções responsáveis por fazer a interface com o usuário, coletando os dados para que sejam enviados para a class diagnosticador e por consequência gerar as preliminares sobre cada IST. Também foi criado nessa parte, a lógica para que seja executado vários diagnosticos de forma paralela.

main.**init_process**(*doencas*, *sintomas*)

Função responsável por criar uma lista com todos os processos que serão executados. Esses processos são os diagnósticos de cada IST. Duas listas são retornadas como resultado dessa função, a primeira nomeada como process possui todos os processos que serão executados e a segunda process_started é uma lista vazia que será povoada quando um processo for iniciado.

Parâmetros

- **doencas** (*list[]*) – lista com o objeto de cada função de diagnóstico IST
- **sintomas** (*Sintomas*) – Objeto da classe Sintomas

Retorna process:

Tipo de retorno list[]

Retorna process_started:

Tipo de retorno list[]

main.**mockaDados**(*sintomas*)

Função responsável por coletar informações do usuário para realizar o diagnostico. Perguntas do tipo, 'qual foi a ultima relação sexual desprotegida' e 'quais sintomas o usuário está sentindo', são abordadas aqui. O objeto 'sintomas' passado por parâmetro será modificado recebendo os valores informado. A função valor() é utilizada como auxiliar.

Parâmetros **sintomas** (*Sintomas*) – Objeto da classe Sintomas

main.**start_process**(*process*, *process_started*)

Função que inicia cada processo de acordo com o limite imposto de execuções paralelas. Após iniciar um processo, este será transferido para outra lista de processos que foram iniciados.

Parâmetros

- **process** (*list[]*) – Essa lista possui todos os processos registrados que estão aguardando para serem executados.

- **process_started** (*list[]*) – Lista com todos os processos que foram executados e que ainda se encontram em execução.

main.**valor**(*description*)

Função para coletar a intensidade do sintoma sentida pelo usuário. Para isso, é usado uma lista de enumeradores contendo todas as intensidades cadastradas no sistema. Após exibir essas opções, é mostrado de qual sintoma se trata. Essa informação é coletada e retornada para onde chamou.

Parâmetros **description** (*String*) – Descrição sobre o sintoma que será coletado a informação.

Retorna intensidade: Intensidade sentida pelo usuário para o sintoma questionado.

Tipo de retorno int

1.2 Módulo Diagnosticador

Nesse módulo, existem 2 classes bases para construir labels de intensidade do sintoma e a respeito da possibilidade de ter a IST. Além disso, foi implementado a class Diagnosticador que abriga o diagnóstico de cada IST abordada pelo sistema.

class diagnosticador.**Diagnosticador**(*in_met_defuzz='centroid'*, *acuracia=0.1*)

Base: object

diagnostico_cancro_mole(*sintomas*)

Método para gerar diagnóstico sobre a IST cancro mole, de acordo com os sintomas informado pelo usuário. O código é feito em alguns passos, são eles:

Passos:

- 1º: Define as variáveis de entrada do sistema, o universo em questão, isto é, o intervalo de valores.
- 2º: Define o conjunto Fuzzy/Função de pertinência para cada variável linguistica escolhida.
- 3º: Define a variável de saída do sistema.
- 4º: Conjunto fuzzy/Função de pertinência para cada saída.
- 5º: Definição das regras do sistema.
- 6º: Cria o cotrolador do sistema.
- 7º: Recebe as entradas, as computa e gera o resultado.

Parâmetros **sintomas** (Sintomas) – Objeto da classe Sintomas

diagnostico_clamidia(*sintomas*)

Método para gerar diagnóstico sobre a IST clamídia, de acordo com os sintomas informado pelo usuário. O código é feito em alguns passos, são eles:

Passos:

- 1º: Define as variáveis de entrada do sistema, o universo em questão, isto é, o intervalo de valores.
- 2º: Define o conjunto Fuzzy/Função de pertinência para cada variável linguistica escolhida.
- 3º: Define a variável de saída do sistema.
- 4º: Conjunto fuzzy/Função de pertinência para cada saída.
- 5º: Definição das regras do sistema.
- 6º: Cria o cotrolador do sistema.
- 7º: Recebe as entradas, as computa e gera o resultado.

Parâmetros **sintomas** (Sintomas) – Objeto da classe Sintomas

diagnostico_gonorreia(*sintomas*)
Método para gerar diagnóstico sobre a IST gonorreia, de acordo com os sintomas informado pelo usuário. O código é feito em alguns passos, são eles:

Passos:

- 1º: Define as variáveis de entrada do sistema, o universo em questão, isto é, o intervalo de valores.
- 2º: Define o conjunto Fuzzy/Função de pertinência para cada variável linguistica escolhida.
- 3º: Define a variável de saída do sistema.
- 4º: Conjunto fuzzy/Função de pertinência para cada saída.
- 5º: Definição das regras do sistema.
- 6º: Cria o cotrolador do sistema.
- 7º: Recebe as entradas, as computa e gera o resultado.

Parâmetros **sintomas** (Sintomas) – Objeto da classe Sintomas

diagnostico_herpes_genital(*sintomas*)
Método para gerar diagnóstico sobre a IST herpes genital, de acordo com os sintomas informado pelo usuário. O código é feito em alguns passos, são eles:

Passos:

- 1º: Define as variáveis de entrada do sistema, o universo em questão, isto é, o intervalo de valores.
- 2º: Define o conjunto Fuzzy/Função de pertinência para cada variável linguistica escolhida.
- 3º: Define a variável de saída do sistema.
- 4º: Conjunto fuzzy/Função de pertinência para cada saída.
- 5º: Definição das regras do sistema.
- 6º: Cria o cotrolador do sistema.
- 7º: Recebe as entradas, as computa e gera o resultado.

Parâmetros **sintomas** (Sintomas) – Objeto da classe Sintomas

diagnostico_hiv(*sintomas*)
Método para gerar diagnóstico sobre a IST HIV, de acordo com os sintomas informado pelo usuário. O código é feito em alguns passos, são eles:

Passos:

- 1º: Define as variáveis de entrada do sistema, o universo em questão, isto é, o intervalo de valores.
- 2º: Define o conjunto Fuzzy/Função de pertinência para cada variável linguistica escolhida.
- 3º: Define a variável de saída do sistema.
- 4º: Conjunto fuzzy/Função de pertinência para cada saída.
- 5º: Definição das regras do sistema.
- 6º: Cria o cotrolador do sistema.
- 7º: Recebe as entradas, as computa e gera o resultado.

Parâmetros **sintomas** (Sintomas) – Objeto da classe Sintomas

diagnostico_sifilis_estagio1(*sintomas*)
Método para gerar diagnóstico sobre a IST sífilis estágio 1, de acordo com os sintomas informado pelo usuário. O código é feito em alguns passos, são eles:

Passos:

- 1º: Define as variáveis de entrada do sistema, o universo em questão, isto é, o intervalo de valores.
- 2º: Define o conjunto Fuzzy/Função de pertinência para cada variável linguistica escolhida.
- 3º: Define a variável de saída do sistema.
- 4º: Conjunto fuzzy/Função de pertinência para cada saída.
- 5º: Definição das regras do sistema.
- 6º: Cria o cotrolador do sistema.
- 7º: Recebe as entradas, as computa e gera o resultado.

Parâmetros **sintomas** (Sintomas) – Objeto da classe Sintomas

diagnostico_sifilis_estagio2(*sintomas*)

Método para gerar diagnóstico sobre a IST sífilis estágio 2, de acordo com os sintomas informado pelo usuário. O código é feito em alguns passos, são eles:

Passos:

- 1º: Define as variáveis de entrada do sistema, o universo em questão, isto é, o intervalo de valores.
- 2º: Define o conjunto Fuzzy/Função de pertinência para cada variável linguistica escolhida.
- 3º: Define a variável de saída do sistema.
- 4º: Conjunto fuzzy/Função de pertinência para cada saída.
- 5º: Definição das regras do sistema.
- 6º: Cria o cotrolador do sistema.
- 7º: Recebe as entradas, as computa e gera o resultado.

Parâmetros **sintomas** (Sintomas) – Objeto da classe Sintomas

diagnostico_sifilis_estagio3(*sintomas*)

Método para gerar diagnóstico sobre a IST sífilis estágio 3, de acordo com os sintomas informado pelo usuário. O código é feito em alguns passos, são eles:

Passos:

- 1º: Define as variáveis de entrada do sistema, o universo em questão, isto é, o intervalo de valores.
- 2º: Define o conjunto Fuzzy/Função de pertinência para cada variável linguistica escolhida.
- 3º: Define a variável de saída do sistema.
- 4º: Conjunto fuzzy/Função de pertinência para cada saída.
- 5º: Definição das regras do sistema.
- 6º: Cria o cotrolador do sistema.
- 7º: Recebe as entradas, as computa e gera o resultado.

Parâmetros **sintomas** (Sintomas) – Objeto da classe Sintomas

diagnostico_tricomoniase(*sintomas*)

Método para gerar diagnóstico sobre a IST tricomoniase, de acordo com os sintomas informado pelo usuário. O código é feito em alguns passos, são eles:

Passos:

- 1º: Define as variáveis de entrada do sistema, o universo em questão, isto é, o intervalo de valores.
- 2º: Define o conjunto Fuzzy/Função de pertinência para cada variável linguistica escolhida.
- 3º: Define a variável de saída do sistema.

- 4º: Conjunto fuzzy/Função de pertinência para cada saída.
- 5º: Definição das regras do sistema.
- 6º: Cria o cotrolador do sistema.
- 7º: Recebe as entradas, as computa e gera o resultado.

Parâmetros **sintomas** (Sintomas) – Objeto da classe Sintomas

class diagnosticador.**LabelDoencas**(*value*)

Base: enum.Enum

An enumeration.

ALTA = 80

BAIXA = 40

MEDIA = 60

MUITO_ALTA = 100

MUITO_BAIXA = 20

NENHUMA = 0

class diagnosticador.**LabelSintomas**(*value*)

Base: enum.Enum

An enumeration.

ALTA = 100

BAIXA = 40

MEDIA = 70

NENHUMA = 0

class diagnosticador.**Sintomas**

Base: object

Classe com atributos que descrevem os sintomas de cada IST.

diagnosticador.**acc**(**args*)