ChatGPT Task (sprih)

- *Code with some details*

C: > Users > mahesh > Downloads > chat.py > chatgpt_response

```python
# install the necessary libraries
import os
import openai
from colorama import Fore, Style

# using the api_key of openai
openai.api_key = "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"

# setting parameters
ANSWER_SEQUENCE = "\nAI:"
QUESTION_SEQUENCE = "\nHuman: "
TEMPERATURE = 0.5
MAX_TOKENS = 500
FREQUENCY_PENALTY = 0
PRESENCE_PENALTY = 0.6

# number of questions we want to add into context
MAX_CONTEXT_QUESTIONS = 10
```

Imported the required libraries here
openai ⇒ for using the api_key and model
os ⇒ for interacting with the terminal
colorama ⇒ for changing the formatting of text in the terminal

↳ This is the open ai api_key through which the model selected works and user is able to interact via terminal.

⇒ These are the variable initiating the sequence as human and AI. These will be further used for creating content.

These are other different parameters of the model.

⟶ In order to generate content in our chat (conversation) we need this.

- *chatgpt_response() function*

```python
# function including all the set parameters for the openai model that is being used
def chatgpt_response(prompt):
    """
    This function takes the prompt given by user as input and returns the response from the model.

    Parameters:
    prompt (str datatype): Input to the function
    Returns the response from the model
    """

    response = openai.Completion.create(
        model="text-davinci-003",
        prompt=prompt,
        temperature=TEMPERATURE,
        max_tokens=MAX_TOKENS,
        top_p=1,
        frequency_penalty=FREQUENCY_PENALTY,
        presence_penalty=PRESENCE_PENALTY,
    )
    return response.choices[0].text
```

The above function will provide us the response that the OpenAI model has generated using the ***API key***.

The model takes different parameters:

- ***model:*** It specifies which GPT model to use. Here ***text-davinci-003*** model is used.
- ***prompt:*** This is the input text used to generate the response.
- ***temperature:*** It controls the creativity of the language model's output. *Higher temperature will give more varied and unpredictable output while lower temperature will give more predictable and Safe output.*
- ***max_tokens:*** sets *maximum limit on the number of tokens* that is generated by the language model.
- ***top_p:*** It controls the randomness of the language model. *Higher value of top_p will give more diverse output while low value gives more predictable and safe output.*
- ***frequency_penalty:*** It is used to restrict the model from generating the same words and phrases repeatedly. *Higher value gives more varied output, low value gives more repetition.*
- ***presence_penalty:*** It restricts the model from generating the words and phrases that are already generated in earlier response. *Higher value gives more varied output, low value gives more repetition.*

- *main() function*

```python
def main():

    # clear the terminal
    os.system("cls")

    print(Fore.MAGENTA + Style.BRIGHT + "Ask your questions here..." + Style.RESET_ALL)
    print()

    # store previous questions and answers to keep track
    previous_questions_answers = []

    while True:

        # question from user
        n_question = input(Fore.GREEN + Style.BRIGHT + "You: " + Style.RESET_ALL)

        # to close the chatbot
        if n_question == "close":
            print(
                Fore.CYAN
                + Style.BRIGHT
                + "ChatGPT: "
                + Fore.MAGENTA
                + "Sure! Have a great day :)"
                + Style.RESET_ALL
            )
            break

        # Building context for next questions
        context = ""
        for question, answer in previous_questions_answers[-MAX_CONTEXT_QUESTIONS:]:
            context += QUESTION_SEQUENCE + question + ANSWER_SEQUENCE + answer

        # adding the new question to the end of the context
        context += QUESTION_SEQUENCE + n_question + ANSWER_SEQUENCE

        # getting response from the function for the input context
        response = chatgpt_response(context)

        # adding new question and answer to the previous_questions_answers list
        previous_questions_answers.append((n_question, response))

        # printing the response
        print(Fore.CYAN + Style.BRIGHT + "ChatGPT: " + Style.NORMAL + response)
        print()


if __name__ == "__main__":
    main()
```

#task