Napisz program, który wylicza przybliżoną liczbę PI metodą całkowania numerycznego całki oznaczonej z funkcji. Program winien mieć możliwość ustawienia przez użytkownika ilości liczb z przedziału całki oznaczonej oraz ustawienia ilości wątków celem zrównoleglenia obliczeń matematycznych. Zrównoleglenie wykonać za pomocą biblioteki „thread" która działa w systemach z standardem POSIX. Program musi być napisany w języku C++. Program powinien wypisywać na ekran terminala czas liczenia całki wraz z wynikiem. Poproś czata o opisanie jak działa algorytm oraz program który wygenerował.

Po napisaniu programu należy przeprowadzić testy algorytmu. W programie ustala się ilość liczb podziału z przedziału całki oznaczonej np. 100'000'000, 1'000'000'000, 3'000'000'000. Następnie uruchamia się program z ilością wątków od 1 do 50. Czasy z pomiarów i liczba wątków zapisać w formie tabelarycznej w Excelu.

Po wykonaniu pomiarów, wyniki zapisane do arkusza kalkulacyjnego pozwalają narysować wykres gdzie oś Y będzie to czas, a oś X będą to wątki (powinniśmy dostać trzy linie na wykresie). Rysunek wykresu wklej do dokumentacji.

Podczas testowania w systemie Windows otwórz menedżer zadań i sprawdzaj zajętość procesora, zajętość rdzeni, ilość wątków, częstotliwość procesora, itp. możesz kilka rysunków wkleić do dokumentacji.

Kolejnym krokiem jest sprawdzenie, czy zmniejsza się czas pracy programu gdy zwiększamy ilość wątków. Jeśli tak to mamy program dobrze napisany. Jeśli nie to powinniśmy szukać błędu algorytmicznego. Zobacz jaki najmniejszy czas dostaniesz przy jakiej ilość wątków. Czy parametry twojego komputera mają znaczenie? Jeśli masz jakiś inny komputer to sprawdź czy te same wyniki dostaniesz.

Celem projektu jest napisanie programu wielowątkowego z wykorzystaniem czata. Podczas pisania programu należy odpowiedzieć na pytania takie jak: jak jest generowany kod (poprawność merytoryczna), czy kod się kompiluje, czy kod się uruchamia, czy nie ma błędów podczas kompilacji, czy nie ma rażących błędów np. wycieków pamięci, ile razy trzeba było generować kod, czy dużo było poprawy kodu, czy poprawne dołączone są biblioteki, czy dostajemy poprawne wyniki (jeśli nie to spróbujmy aby czat znalazł błąd i go poprawił). Jeśli dalej wyniki wychodzą błędnie to spróbuj aby czat przedstawił jakie mogą być przyczyny źle działającego programu. Przy problemach z uruchomieniem programu skorzystaj z podpowiedzi które będzie sugerowała AI.

Po napisaniu programu zastanów się i odpowiedz na pytania: jakie są zagrożenia przy korzystaniu z AI, czy należy korzystać z AI przy pisaniu programów, w czym AI pomaga, a w czym przeszkadza. Jak muszą być definiowane polecenia aby AI poprawnie wygenerowała kod, czy może programować osoba nieznająca się na programowaniu (na chwilę obecną), jaka jest różnica między czatem a GitHub Copilot? Poprzedni projekt był z zastosowaniem GitHub Copilot więc wiesz jak się z niego się korzysta oraz jakie ma wady i zalety. Można spróbować generować program z użyciem innych czatów lub wersji (tylko w dokumentacji napisz który to czat). Można w dokumentacji zaprezentować ciekawe dialogi prowadzone z AI. Zastanów się jak może wyglądać przyszłość AI.

Do projektu należy napisać dokumentację w Latex, DoxyGen oraz zapisać projekt w serwisie GitHub. Chciałbym aby w dokumentacji pochylić się nad takimi przykładowymi pytaniami jak są

powyżej. Oczywiście jeśli masz jakieś inne pytania lub sugestie odnośnie AI to napisz je.
W dokumentacji nie powinno zabraknąć wykresu oraz tabelek z arkusza kalkulacyjnego wraz z opisem wyników działania programu. Program piszemy jednoosobowo.