PZ FM NI – zadania do poćwiczenia przed 1. kolokwium

1. Poproś użytkownika o wpisanie z klawiatury 3 liczb (umów się na jakiś typ zmiennych). Następnie wypisz na ekran największą i najmniejszą z nich.

2. Twój program ma dla podanego *x* wypisać N pierwszych wyrazów szeregu Taylora dla funkcji f(x) = exp(x) , który ma postać $f(x) \approx \sum_{i=0}^{N} \frac{x^i}{n!}$:

- W funkcji main wczytaj od użytkownika z klawiatury x oraz n. Zapewnij, aby n było całkowite.
- Zakoduj i sprawdź funkcję double silnia (double x) .
- Zakoduj i wywołaj w main funkcję double MyExp (double x, int N) , która wykona powyższą pętlę. Dla przejrzystości kodu, rozważ utworzenie osobnej zmiennej do składowania sumy kolejnych wyrazów, a osobnej - do umieszczenia wartości bieżącego wyrazu.
- W main() wypisz wynik na ekran.

Wzbogać teraz program o „bezpiecznik" – zmienną epsilon, mieszczącą najmniejszą wartość, jaką dany obliczony wyraz może mieć. Gdy wyraz będzie mniejszy od epsilonu, pętla ma się zatrzymać, a funkcja ma zwrócić wyliczoną do tego kroku sumę. Pozwól, aby to użytkownik w funkcji main podawał ten epsilon z klawiatury.

3. Z balonu wiszącego na wysokości y = 12000 m upuszczono paczkę o masie m = 10 kg. Paczka spada w polu grawitacyjnym o współczynniku g = 9.81 m/s^2. Twoim zadaniem jest wykonanie symulacji spadku paczki w krótkich odstępach czasowych co dt = 1 s, zgodnie z równaniami Newtona. Celem jest sprawdzenie prędkości zderzenia paczki z ziemią (y = 0), w dwóch wariantach:

(a) najpierw – przy braku powietrza

(b) następnie pod obecność powietrza. Przyjmij siłę oporu jako:

$$F_{op}(v) = -\frac{1}{2} C \cdot \rho \cdot S \cdot v^2 ,$$

gdzie C = współczynnik oporu powietrza (przyjmij 0.70)
ρ = gęstość powietrza (przyjmij 1.21 kg/m^3)
S = przekrój poprzeczny ciała (przyjmij 0.50 m^2)
v = prędkość *chwilowa* ciała

Wskazówki:

- w każdym kroku pętli obliczaj kolejne siły, a stąd – przyspieszenie a.
- Na tej podstawie aktualizuj prędkość, zmieniając po chwili Δt o: Δv = a · Δt
- Teraz aktualizuj położenie y, zmieniając po chwili Δt o: Δy = v · Δt
- Jeśli nie jesteś pewien znaku sił, przyspieszenia, prędkości - rozrysuj sytuację.

4. Paraboloida obrotowa to powierzchnia w $\mathbb{R}^3$, opisana równaniem $Z = x^2 + y^2$. Paraboloida została obcięta od góry płaszczyzną Z = 1 (uwaga: brzeg obcięcia jest okręgiem o promieniu 1). Korzystając z metody losowania Monte-Carlo, wyznacz przybliżoną objętość bryły pod tak obciętą paraboloidą. Wykonaj np. 10^7 losowań.
Wskazówki: zbiór losowanych [x, y] nie powinien tworzyć kwadratu, a koło o promieniu 1.
Oczekiwany wynik = π/2.

5. Rozważ poniższą funkcję w przedziale X ∈ [-5, 5], Y ∈ [-5, 5] i zapoznaj się z jej kształtem np. w środowisku Desmos (można obracać wykres i zoomować jego zakres) :

$$f(x) = \frac{\sin(x-1)}{x-1} \cdot \cos(y) \cdot \left(1 + \frac{y}{10}\right)$$

Zakoduj f(x) w funkcji o nagłówku double fun (double x, double y). Ale zauważ, że x = 1 ∉ do dziedziny f(x). Zabezpiecz tę funkcję tak, że gdyby wywoływać ją dla *x* = 1, to miejsce czynnika z iksami zająć ma wartość 1 (kierujemy się tym, że $\lim_{x \to 0} \sin(x) / x = 1$) .

Zakoduj funkcję void FindMin, przyjmującą metodą referencji 3 zmienne typu double: Xmin, Ymin i Zmin. Funkcja ma przejść po wartościach przedziału z krokiem np. 0.005 i znaleźć położenie (Xmin, Ymin) globalnego minimum oraz jego wartość Zmin.
W efekcie, po zakończeniu funkcji FindMin, ten punkt zostanie zwrócony (metodą referencji). W main wypisz współrzędne znalezionego punktu.

Dla chętnych: pomyśl, jak prostym algorytmem można wychwycić każde z minimów lokalnych. Wzbogać kod FindMin tak, aby dodatkowo zliczała ilość lokalnych minimów na badanym przedziale. Rozszerz nagłówek FindMin o dodatkową referencję na zmienną typu int, przez który zwrócisz tę ilość lokalnych minimów. Zmodyfikuj main, aby dodać nową zmienną (np. int N_local_minima = 0) i użyj jej w wywołaniu FindMin, oraz dodaj ją do wypisów na ekran.

6. Przepisz zadanie 5 w taki sposób, aby mechanizmem przekazywania zmiennych nie były referencje, a wskaźniki.

7. Droga po krzywej zadanej parametrycznie (tj. X = X(t) , Y = Y(t)) wyraża się wzorem całkowym, który można przybliżyć do sumy po małych elementach Δt:

$$l = \int_{t_1}^{t_2} \sqrt{\left[\frac{dX(t)}{dt}\right]^2 + \left[\frac{dY(t)}{dt}\right]^2} \, dt \approx \sum_{t_1}^{t_2} \sqrt{\left[\frac{dX(t)}{dt}\right]^2 + \left[\frac{dY(t)}{dt}\right]^2} \cdot \Delta t$$

Wiemy, że okrąg zadany jest parametrycznie przez: [X(t) = sin(t) , Y(y) = cos(t)] . Możesz to sprawdzić np w środowisku Desmos. Potrzebna we wzorze na l pochodna wygląda więc tak: [dX/dt = cos(t) , dY/dt = -sin(t)] .

Twoim zadaniem jest obliczenie drogi po górnej połówce okręgu (wynik: π) . Zatem:

- Utwórz funkcje double dXdt (double t) i odpowiednio dYdt, gdzie wpiszesz wzory.
- Utwórz funkcję double PathLength (double t1, double t2, double dt) i zakoduj w niej algorytm „całkowania" (sumowania) drogi po t1 $\in$ [t1 ... t2] z krokiem dt, według wzoru powyżej.
- W funkcji main wywołaj PathLength w przedziale [-π/2, +π/2] z krokiem np. 1e-5. Wypisz wynik na ekran.

8. Przepisz zadanie 5 (lub 6) tak, aby funkcja FindMin – zamiast mieć wpisane jawnie wywołania funkcji fun, przyjmowała wskaźnik funkcyjny f(x) jako dodatkowy argument i wywoływany byłby ten wskaźnik. Za to w main, w miejscu wywołania FindMin, należy dodatkowo podać nazwę funkcji fun.

9. Średnia $\langle x \rangle$ z N wyników x_i ważona ich niepewnościmi Δx_i wyraża się poniższym wzorem. Wzór na prawo od niej opisuje niepewność $\Delta\langle x \rangle$.

$$\langle x \rangle = \left. \sum_{i=1}^{N} \frac{x_i}{\left(\Delta x_i\right)^2} \middle/ \sum_{i=1}^{N} \frac{1}{\left(\Delta x_i\right)^2} \right. \qquad \Delta\langle x \rangle = \sqrt{\frac{1}{\sum_{i=1}^{N} \frac{1}{\left(\Delta x_i\right)^2}}}$$

Szereg grup badało czas życia neutronu. Ich wyniki z niepewnościami podaje ta tabela:

T [s]	877.75	878.3	877.7	881.5	880.2	882.5	880.7	878.5
Δ*T* [s]	0.28	1.6	0.7	0.7	1.2	1.4	1.3	0.5

Utwórz na tej podstawie tablice z wynikami i niepewnościami. Następnie, oblicz $\langle x \rangle$ oraz $\Delta\langle x \rangle$ i wypisz wyniki na ekran.