MJUKVARULABORATION 1 i Datorsystemteknik.

Uppgift 1) - Enklare funktion, loopar, villkor.

a) Skriv kod för att uppskatta PI.
Ett sätt att uppskatta PI är att gör det med summan:

PI_est = 4 x (1 – 1/3 + 1/5 – 1/7 + 1/9 - …….)

Beräkna PI_est upp till 1/99.

b) Tag ovanstående kod och skapa en funktion:

```
double PI_estimator ( int nMax )
```

som returnerar PI_est ovan, samt tar `nMax` som argument (motsvarande 99 i uppgiften ovan).

c) Använd funktionen ovan för att bestämma hur lång summan behöver vara (hur stort `nMax` behöver vara) för att precision blir ± 1% av 3.14159.

Uppgift 2) - Enklare funktion, loopar samt villkor.

I Appendix 1 listas 100 tal i en vektor. Skriv funktioner för att beräkna följande egenskaper hos vektorn:

1) Medelvärdet
2) Minsta värdet
3) Största värdet
4) Variansen.

Medelfelet

Färdigställ funktionen `mean(..)` så att den beräknar medelvärdet av alla element i vektorn `dVec` och returnerar detta medelvärde. Parametern `nVals` anger hur många element som finns i vektorn `dVec`.

```
double mean(double dVec[], int nVals);
```

Ledning: Medelvärdet beräknas som summan av alla element i vektorn dividerat med antal element i vektorn, enligt ekvationen:

$$E_{Average} = \frac{1}{N}\sum_{i=1}^{N} dVec(i) \qquad \text{(Medelfelet)}$$

Minsta värdet

Färdigställ funktionen `minima(..)` så att den letar upp och returnerar det minsta av alla element i vektorn `dVec`. Parametern `nVals` anger hur många element som finns i vektorn.

```
double minima(double dVec[], int nVals);
```

Största värdet

Färdigställ funktionen `maxima(..)` så att den letar upp och returnerar det största av alla element i vektorn `dVec`. Parametern `nVals` anger hur många element som finns i vektorn.

```
double maxima(double dVec[], int nVals);
```

Variansen

Färdigställ funktionen `variance(..)` så att den beräknar variansen av alla element i vektorn `dVec` och returnerar denna varians. Parametern `nVals` anger hur många element som finns i vektorn. Om det inte går att beräkna variansen skall denna funktion returnera värdet –1.0, t.ex. om antal element i vektorn är färre eller lika med 1.

```
double variance(double dVec[], int nVals);
```

Ledning: Variansen beräknas som summan av alla element minus deras medelvärde i kvadrat i vektorn dividerat med antal element i vektorn minus ett, enligt ekvationen:

$$E_{Variance} = \frac{1}{N-1}\sum_{i=1}^{N} (dVec(i) - E_{Average})^2 \qquad \text{(Variansen)}$$

Uppgift 3) - Enklare funktion, loopar, vilkor, arrayer, egen datatyp, storlek på datatyp.

a) Skapa en funktion enligt
`stats_vars calc_stats(double dVec[], int nVals);`
där
`stats_vars`

är en egen datatyp som innehåller följande:
1) antalet datapunkter
2) medelvärdet
3) min-värdet
4) max-värdet
5) variansen

Ledning: Nyckelord är `typedef` och `struct`.

Visa att funktionen fungerar genom att testa med vektor av tal i Appendix 1.

b) Kontrollera storleken hos variabeln ni skapat.

Ledning: Nyckelord är `sizeof`.

Jämför om storleken på den nya datatypen är samma som summan av delarna.

ALLMÄNA TIPS:

Det finns en funktion man kan använda (om man vill) för att beräkna kvadraten på x.

```
pow(x, y)
```

där x är värdet och y är det man upphöjer x i (exempelvis 2 vid kvadraten).

För att nyttja funktionen skall man använda mattebiblioteket i C genom att skriva följande högst up i källkodsfilen.

```
#include <math.h>
```

EXTRA UPPGIFTER (valfria):

- Kontrollera vad den inbyggda slumptalsfunktionen `rand()` returnerar för medelvärde och varians.
- Skapa en egen funktion normerar en vektor av tal till medelvärde = 0.0 och standardavvikelse 1.0 (roten ur, `sqrt()`, variansen).
- Lista storleken på grundtyperna i C dvs `char, short, int, long, float, double, longdouble`.
- Testa om man skapar en egen datatyp med blandade storlekar och se om summan alltid blir summan av delarna.

APPENDIX 1:

0.9501, 0.2311, 0.6068, 0.4860, 0.8913, 0.7621, 0.4565, 0.0185, 0.8214, 0.4447, 0.6154,
0.7919, 0.9218, 0.7382, 0.1763, 0.4057, 0.9355, 0.9169, 0.4103, 0.8936, 0.0579, 0.3529,
0.8132, 0.0099, 0.1389, 0.2028, 0.1987, 0.6038, 0.2722, 0.1988, 0.0153, 0.7468, 0.4451,
0.9318, 0.4660, 0.4186, 0.8462, 0.5252, 0.2026, 0.6721, 0.8381, 0.0196, 0.6813, 0.3795,
0.8318, 0.5028, 0.7095, 0.4289, 0.3046, 0.1897, 0.1934, 0.6822, 0.3028, 0.5417, 0.1509,
0.6979, 0.3784, 0.8600, 0.8537, 0.5936, 0.4966, 0.8998, 0.8216, 0.6449, 0.8180, 0.6602,
0.3420, 0.2897, 0.3412, 0.5341, 0.7271, 0.3093, 0.8385, 0.5681, 0.3704, 0.7027, 0.5466,
0.4449, 0.6946, 0.6213, 0.7948, 0.9568, 0.5226, 0.8801, 0.1730, 0.9797, 0.2714, 0.2523,
0.8757, 0.7373, 0.1365, 0.0118, 0.8939, 0.1991, 0.2987, 0.6614, 0.2844, 0.4692, 0.0648,
0.9883