Name and student number: ____

PSCB57. MIDTERM EXAM 21 OCT 2019 - PROBLEMS WITH **SOLUTIONS**

Points in the square brackets give the idea of relative weight of problems. This part is worth **11% of the total course score**. Use left (blank) pages of the booklet as scratchpad and mark it as such. Write legibly a well commented Python program on the right, lined, pages. In the program header, please explain in words the scheme of your code, i.e., what your algorithm is doing step by step. For instance: *"Program to find a maximum of (...). First reads the input from keyboard. Then it scans the values of unknown Z in (..) range, dividing search range into N equal intervals. Using a 'for' loop from 1 to N, program iteratively calls function (..) computing (..) and compares the returned value of variable (..) with the stored value in (..) to find a maximum of (..) etc.* You can describe the algorithm separately from the code if you wish, but the lines of code still must have brief comments on what they are supposed to do. Even if you get stuck somewhere, you may get a partial credit. We value more a clear and correct algorithm than correct Python syntax or style.

1 [5 pts.] Walks in 2 dimensions

Write a function to perform one random walk consisting of N steps, starting from value (x,y)=(0,0), and increasing x and y in every step by (dx,dy). To calculate the random variables dx and dy, use numpy random number generator s = np.random.choice([-2,-1,1,2]), which randomly (with equal probability) stores -2, -1, +1, or +2 in variable s. Return to the calling program the final square of the distance from the starting piont, $d^2 = x^2 + y^2$.

In the calling program, set N=100. Write a code to call the above function M = 50 times and print two columns, one numbering the walks, and the other listing final distance from the starting point. Also compute and print the arithmetic mean distance $d_m = < d >$.

1.1 Solution

Of course there are going to be numerous detailed approaches, all of them correct, so here is one of them

```
import numpy as np
'''
   M=50 random 2-D walks of N=100 steps each.
   The length of step in both x and y is equally likely 1 or 2 (no zero-steps)
   Steps in x and y direction are independent and random.
'''

# this function performs one random walk of N steps in 2 dimensions
def walk(N):
    x,y = (0,0) # starting from the origin
    for n in range(N):  # N steps
        x += np.random.choice([-2,-1,+1,+2])
        y += np.random.choice([-2,-1,+1,+2])
    return(x*x+y*y)
```

```
# main program
# calls function doing one walk of N random walk steps M times
N, M, i, d_aver  = 100, 50, 0, 0.
print("\ni      d")  # print header of 2-column output
while(i<M):
    i += 1
    d2 = walk(N)   # distance squared
    d = d2**0.5    # final distance of this walk
    d_aver += d    # compute average distance
    print(i,' ',d) # print 2 columns
print('\n Average distance <d> =',d_aver/M)
```

The output of this code looks like a 2-column table, followed by something like

$< d > = 21.7337...$

(you can get 19.3989..., or 23.22012..., after all it's a Monte Carlo simulation with M and N which are not too large. Every time the script is run, Pyhon3 generates new pseudorandom numbers).

COMMENT: By the way, if you are interested in predicting the value of average distance, it's easier to adopt a different metric: $\sqrt{< d^2 >}$ (not the $< d >$ - that one is easier to compute, so it was chosen for the exam problem). Since variances sum up linearly with the number of steps (and sub-steps in independent directions), you will be able to quickly find the expected mean-square distance in these random walks from the variance or standard deviation of the pseudorandom number generated as an equally probable choice of values {-1,-1,1,2}:

$\sqrt{< d^2 >} = 22.3606....$

2 [6 pts.] The nylon curtain

You want to buy fabric for a sinusoidal, curving curtain. Its shape in (x,y) coordinates is $y(x) = A\sin(12\pi x/W)$, where $W = 6$ m is the x-span of the curtain, and $A = 0.4$ m is the semi-amplitude of the wave. Compute the required length L of the curtain. Divide the curve into a very large number N of segments of equal x-span (dx=W/N). Within each segment the shape of curtain can be taken as straight line segment. Function $y(x)$ and Pythagoras theorem will allow you to derive and sum the length of N segments. Preform the summation in a loop, updating the value of L.

Comment on whether you could speed up your code by noticing that: (i) the curtain waves periodically a whole number of times between its ends, and (ii) the endpoint of one section is the beginning of the next one.

2.1 Solution

```
'''
   Curtain is devided into a huge number N of segments and its length L
   is summed numerically in a loop from segment lengths dL (given by
   Pythagoras theorem).
```

```
    This implementation is not the fastest possible!
    For efficiency, we could: use numpy arrays, realize that the
    curtain makes 24 identical quarter-waves between its ends
    (only one of which needs to be computed), constant "const" should
    be computed once in main program not in function, etc. However,
    this is one of the simplest and clearest algorithms one can write.
'''
def y(x):
    const = 12*np.pi/W
    return (0.4*np.sin(const*x))

# main program for curtain problem
# x changes from 0 to W=6 m.
# y(x) is the sideways deflection of the curtain

N, W, L = 100000, 6., 0.
dx = W/N
for i in range(N):
    x = i*dx        # x-coordinate of the left end of i-th segment
# y0, y1 are values at the begin and end of i-th segment of curtain
    y0, y1 = y(x), y(x+dx)
    dL = ( dx**2 + (y1-y0)**2 )**0.5  # says Pythagoras
    L = L + dL
print('You need to buy',np.around(L,9),'m of fabric for the curtain.')
```

The printout from this code is:

"You need to buy 11.711392631 m of fabric for the curtain".

So the curtain is almost two times longer than W=6 m.

If you had a roof covered by a corrugated tiles, which needs to be re-surfaced or painted, or is you wanted to see how much larger is the actual area of the wave-covered lake than the still-water lake (important for oxidation? cooling rate? surface tension energy?), then in all such applications you may need to evaluate the length of a particular sinusoid. Use the current scriptm or delve into..

2.2 Mathematics of curve length and elliptic integrals

It is interesting to note that the length of a sinusoid of arbitrary ratio of amplitude A to wavelength λ cannot be written in terms of elementary or the most commonly used special functions available on calculators. (Python is much more than a simple calculator and with the help of a special function module scipy.special it will be able to do the job, as we'll see below.)

I'll use this opportunity to first show you a little bit of math that will allow you to calculate the length of any smooth function. In physics we use line integrals along different trajectories and surfaces. To begin with, realize that

$$dL^2 = dx^2 + dy^2 = [1 + (dy/dx)^2]\, dx^2.$$

In our case, $y(x) = A\sin 2\pi x/\lambda$, where $\lambda = 1$m is the wavelength of the curtain, while the derivative is equal $dy/dx = (2\pi A/\lambda)\cos 2\pi x/\lambda$.

The length of function graph from 0 to W, in general, is equal to

$$L = \int_0^W dL = \int_0^W \sqrt{1 + (dy/dx)^2}\, dx$$

In our case $W = 6$ m, and we have 24 quarter-wavelengths along our curtain. Introducing new constant $B = 2\pi A/\lambda$, and a new variable $\phi = 2\pi x/\lambda$, both non-dimensional, we can simplify the derivative as $dy/dx = B\cos\phi$, and write length L as

$$L = (12\lambda/\pi) \int_0^{\pi/2} \sqrt{1 + B^2\cos^2\phi}\, d\phi.$$

The integral will transform into a complete elliptic integral of the second kind $E(k)$, if we change cosine to sine function via identity $\cos^2\phi = 1 - \sin^2\phi$:

$$L = (12\lambda/\pi) \int_0^{\pi/2} \sqrt{1 + B^2 - B^2\sin^2\phi}\, d\phi = (12\lambda\sqrt{1+B^2}/\pi) \int_0^{\pi/2} \sqrt{1 - k^2\sin^2\phi}\, d\phi,$$

where $k^2 = B^2/(1+B^2)$, which with $B = 2\pi A/\lambda = 2.5132741228718345$ gives $k = B/\sqrt{1+B^2} = 0.929152033578$.

$$L = (12\lambda\sqrt{1+B^2}/\pi)\, E(k) = (12\lambda\sqrt{1+B^2}/\pi)\, E(B/\sqrt{1+B^2})$$

can now be computed using for $E(k)$ a calculator at http://keisan.casio.com/exec/system/1180573458,

$$E(k) = 1.1335067013..; \quad L = 11.711392688..\text{m}$$

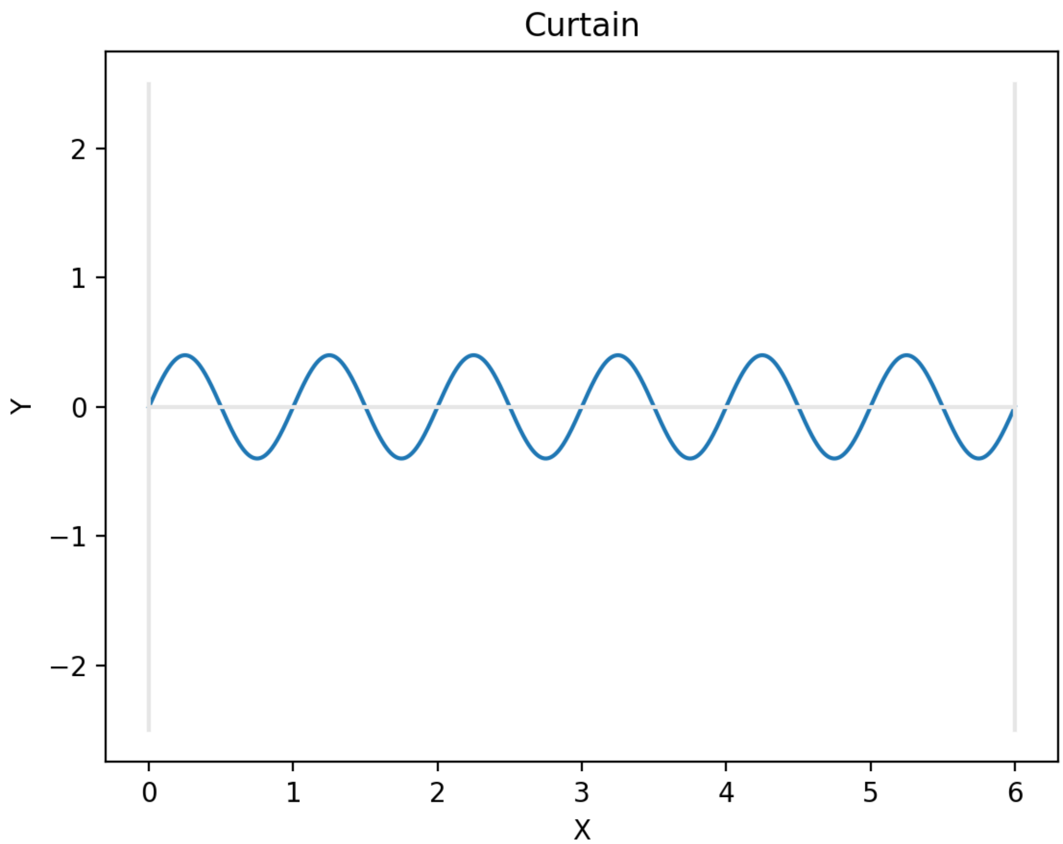
or the SciPy module for special function ellipe(m) = ellipe(k^2) (careful, some software uses k and some k^2 as argument of E function!):

```
>>> from scipy.special import ellipe; import numpy as np
>>> B2 = (2*np.pi*0.4)**2
>>> k2 = B2/(1+B2)
>>> k2**0.5
>>> 0.9291520335781389
>>> ellipe(k2)
>>> 1.1335067005270874
>>> ellipe(k2)*12*(1+(2*np.pi*0.4)**2)**0.5/np.pi
>>> 11.711392680436859
```

This coincides with the numerical integration using N=100000 sectors to *9 accurate digits*! Numerical integration is in this case the method of choice, because of simplicity and accuracy. For more on the length of sinusoids please see discussion on

https://math.stackexchange.com/questions/45089/what-is-the-length-of-a-sine-wave-from-0-to-2-pi

Although plotting was not required, here it is to visualize the curtain:



Comment: The "Nylon Curtain" in the title of this problem is a plagiarized title of an old album by singer Billy Joel, a wordplay on the "Iron Curtain".

PSCB57 Midterm Exam 2019. QUIZ - **SOLVED**. Circle Y[es] or N[o] and submit this page.

This part of the final exam is worth up to **11%** of the total course score. Statements are sometimes be tricky, so read carefully. A sigle word or number may be incorrect. **Any "[N]" answer circled MUST have at least one wrong word circled for credit.** Please disregard typos. Raise you hand if you find something worrying in the text, we will try to answer your question during exam. Programming questions refer to Python ver.3. To account for unintended ambiguity of questions 2 points will be added to your result. Good luck!

[N] An 8-byte integer can assume one of **10**64** different values

[N] Charles Babbage **has built** his Universal Calculator using steam power in the 19th century. It was the first working device based on **binary digit** representation of numbers.

[N] Zuse Z3 computer is considered the first implementation of Boolean or binary arithmetic to computers. Z3 used thousands of **vacuum tubes**.

[N] The first mechanical computer (Antikythera mechanism) performing **4 operations (+,-,*,/)** was created in Sicily at the workshop of famous Greek scientist Archimedes in the 3rd century BC.

[N] Iteration can be done in Python using three different types of loops: 'for' loop, 'while' loop, and **'if' loop**

[N] At a bandwidth of 32 Mbits/s, the network can download 4.8 GB of data in about **4800/32 seconds** = 150 s = 2.5 min.

[N] If tabA is defined as tabA = 2 + np.zeros((2,10),dtype=int), then savetxt("some-io.dat",tabA.T) creates a file consisting of 10 lines of 2 **integers** equal 2 in each line, separated by a **comma**.

[Y] ENIGMA was a German coding machine. Its decoding by Polish cryptologists before World War II, and construction of increasingly sophisticated electromechanical computers called "Bombes" has influenced the outcome of the war and saved countless lives.

[N] After we throw 100 million random points on unit square to evaluate pi by MteCarlo, we expect the accuracy of at least **7 correct digits** of pi=3.1415926..

[N] If ran = np.random.rand(128) and D013 = (ran >= 0.13), then D013 is an array of 128 **floating point numbers**, each between 0.13 and 1.

[Y] The result of operation 1024//10*10 is 1020

[Y] One can put many statements into 1 line of Python code, separated by ";"

[Y] The switch from mainframes to interactive minicomputers in the 1970s became possible thanks to innovations by DEC (Digital Corp.). It led the way to interactive personal computer revolution in 1980s.

[Y] Currently, CPUs consist of many billions of transistors and other microscopic electronic devices inside a microprocessor.

[Y] You need to ask $\sim 10^3$ random people in a pre-election poll, to estimate the outcome of presidential election with statistical error of order 3%. The same number is needed to calculate π by MteCarlo with 3% accuracy

[Y] The first high level language from late 1950s, which still is in wide use, is Fortran. C language became popular in 1970s.

[N] This loop: x = np.linspace(3.,8.,6); for i in x: print(i-2) will print a column of of these numbers: **1 2 3 4 5 6**

[N] Instruction print((-4)**0.5) will print **two values** of square root of -4: 0+2j and 0-2j.

[N] **Only one** of the loop types (the 'for' loop) can be discontinued by the 'break' statement. In a 'while' loop **we should use 'pass'**.

[Y] B = [2,3,4,5]. The following loop: for x in B: print(x**2*2-1) will print a column: 7 17 31 49

[N] The smallest positive floating point number distinct from zero is called **machine epsilon**, and in Python equals about 2.2e-16, because **Python uses 16 bytes** to represent floating point numbers

[N] Python evaluates expression (8e-1 == -1+8e) as **True**

[Y] I have signed the front page of exam with my name and student number.