

Exercises

FIZ 241E

Sym&Num

Q1

if $n > 1$

$m = 10$

else

$m = 20$

- $n = 7, m = ?$
- $n = 0, m = ?$
- $n = -10, m = ?$

Q2

if $T < 30$

$h = 1$

elseif $T < 10$

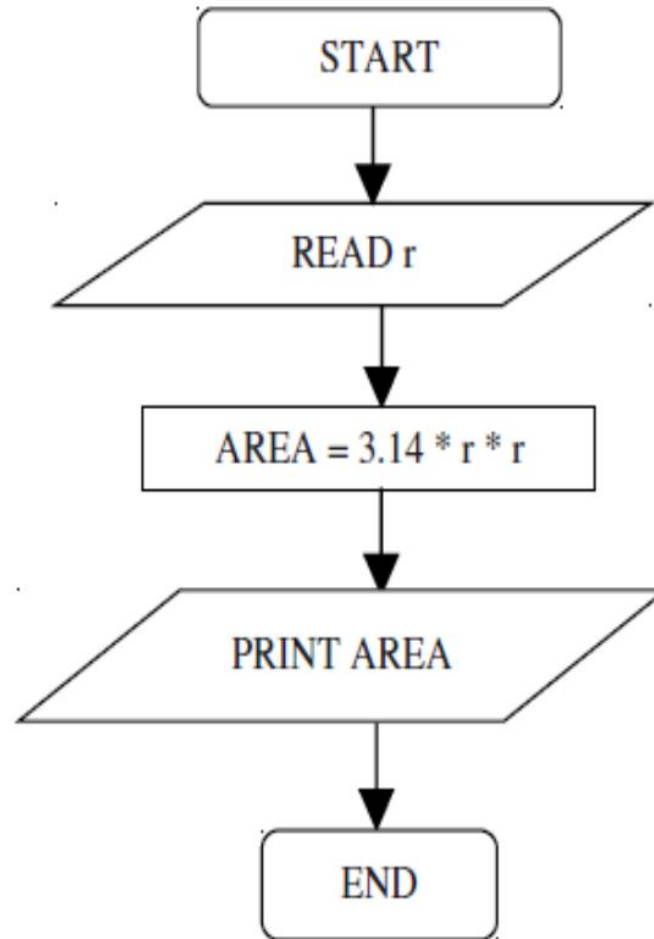
$h = 2$

else

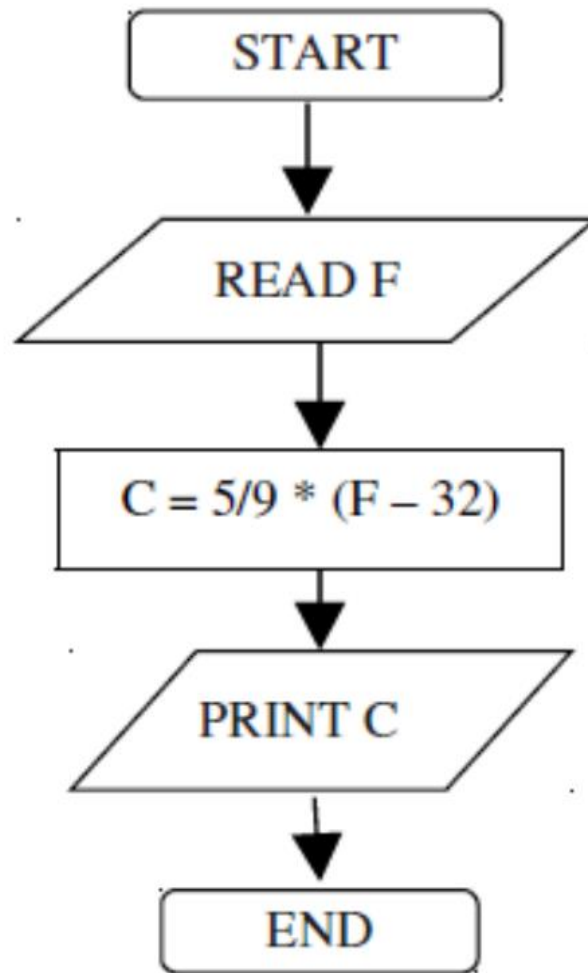
$h = 3$

- $T = 50, h = ?$
- $T = 9, h = ?$
- $T = 0, h = ?$

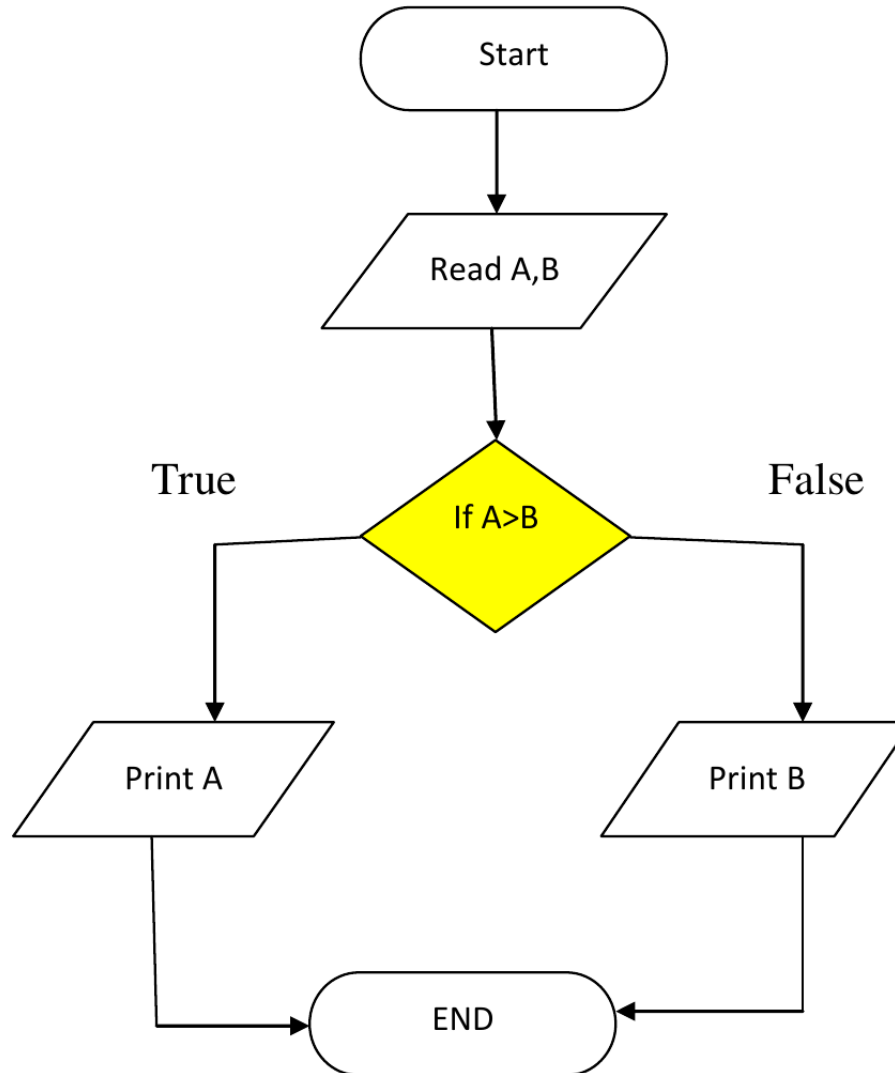
Q3



Q4



Q5



Q6

Write some lines of code using a “for” loop that will print the numbers between 1 and 10 on the screen.

Q7

Write some lines of code that will calculate the sum $1+2+3+\dots+10$.

- The idea is to create a variable that will store the current value of the sum. Set it to zero, then use a for loop that will run through the numbers 1, 2, 3, ... and add each of these to the current sum.
- After the for loop, display the value of the sum.

Q8

Write some lines of code that will calculate the product $1*2*3*...*10$.

- The idea is to create a variable that will store the current value of the product. Set it to one, then use a for loop that will run through the numbers 1, 2, 3, ... and multiply each of these to the current product.
- After the for loop, display the value of the product.

Q9

Write some lines of code to calculate the sum

$$1^2 + 2^2 + 3^2 + \dots 10^2.$$

Use a loop!

Q10

Write some lines of code to calculate the sum
 $1*2+2*3+3*4+ \dots + 9*10$.

Use a single loop!

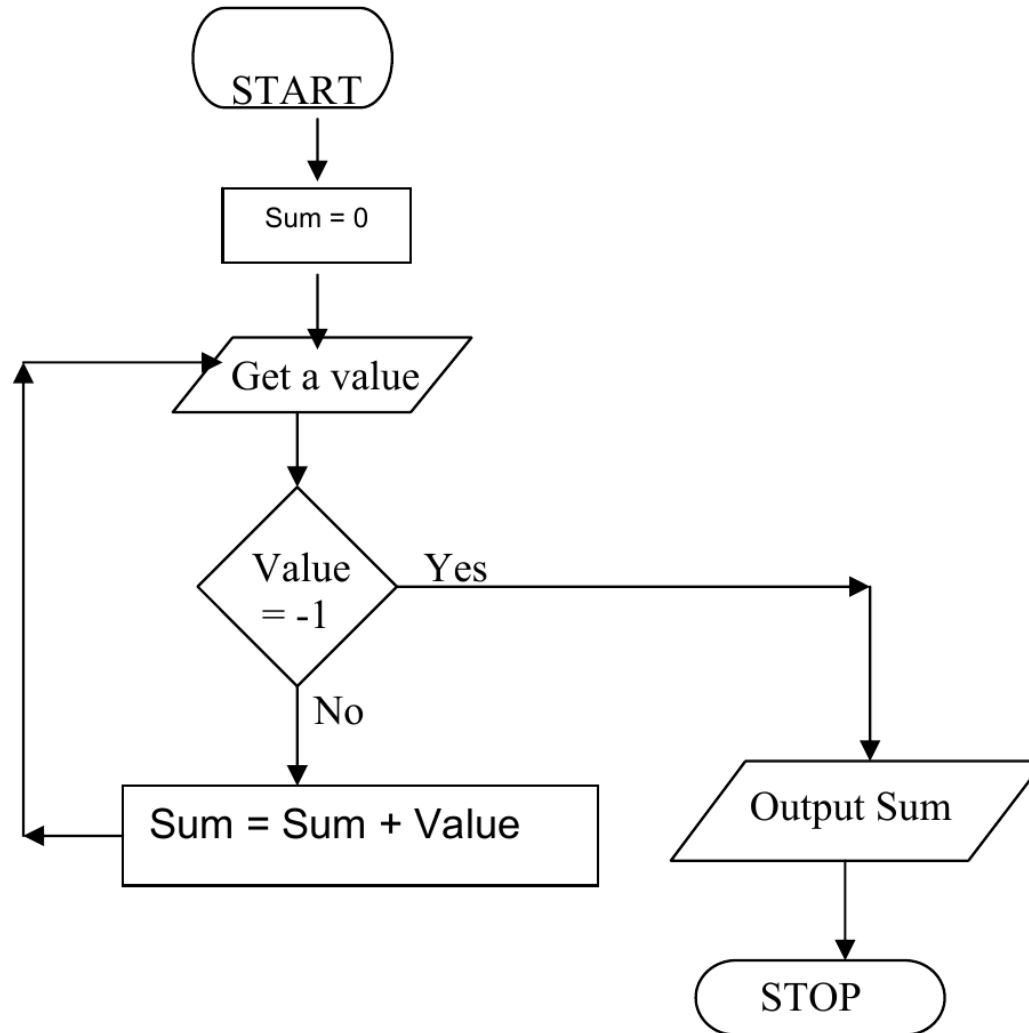
Q11

Write a program to calculate

$$1^1 + 2^2 + 3^3 + 4^4 + 5^5.$$

Use a single loop!

Q12



Q13

Ask user to enter an integer number.

-If the number is 666, print "I do not like this number"
and halt execution

else:

- Print the square of the number if it is odd,
- Print the square root of the number if it is even.
- Then start over by asking a number until the user enters the number 666.

Q14

Write some lines of code that will find how many terms in the sum $1+2+3+\dots$ it requires for the sum to exceed 100.

- The idea is to create a variable that will store the current value of the sum and another variable that keeps track of what number you are adding to the sum.
- **Use a “while” loop** to add the current number to the sum, then increase the current number by 1.
- The while loop should stop once the sum exceeds 100, then display the current number.

Q15

Fibonacci numbers can be calculated by the formula, $F_n = \frac{x^n - (x - \sqrt{5})^n}{\sqrt{5}}$ where $x = \frac{1 + \sqrt{5}}{2}$. Write a program to calculate the n^{th} Fibonacci number (F_n). Your program should take n as an input and print the n^{th} Fibonacci number.

Fibonacci numbers:

0th: 0,

1st: 1,

2nd: 1,

3rd: 2,

3, 5, 8, 13,...

Q16

Let $x = [2, 5, 1, 6]$. x is a vector.

- Add 16 to each element.
- Compute the square root of each element.
- Compute the square of each element.
- Print your results on the screen.

(All of your results should be vectors.)

Q17

Given the vector $x = [1, 8, 3, 9, 0, 1]$, create a short set of commands that will

- add up the values of the elements (the result is a number)
- compute the sine of the given x-values (the result is a vector)

Q18

Think of a population living in an environment with an epidemic disease. A simple model (The SIR Model) can be given as the following:

S: Susceptible (healthy but tends to get the disease)

I: Infected (got the disease, can contaminate S's)

R: Recovered (healthy and cannot be infected again)

At time t we have,

$$S_{t+1} = S_t - a * I_t * S_t$$

$$I_{t+1} = I_t + a * I_t * S_t - b * I_t$$

$$R_{t+1} = R_t + b * I_t$$

where,

a: Contact rate (the probability of getting the disease in a contact between a susceptible and an infectious subject)

b: The rate of recovery

Start at $t=0$, where $S=0.99$ (99%), $I=0.01$ (1%), $R=0.00$ (0%).

Use $a=0.7$ and $b=0.2$,

and simulate 50 days. (Take the time increment as 1 day.)

Show (S-t), (I-t), (R-t) graphs all in the same plot. (Only in MATLAB)

Q19

Fibonacci numbers can be defined as

$$F_n = \begin{cases} 0, & \text{for } n = 0 \\ 1, & \text{for } n = 1 \\ F_{n-2} + F_{n-1}, & \text{for } n > 1 \end{cases}$$

Write a program to calculate the n^{th} Fibonacci number (F_n). Your program should take n as an input and print the n^{th} Fibonacci number.

Q20

What is the output of the following program? The number N is entered as 5.

```
#include<iostream>
using namespace std;
int myfunc1(int num)
{
    int i,mysum=0;
    for(i=0;i<num;i++)
    {
        if (i%2==0) {mysum=mysum+i;}
    }
    return (mysum);}
int main()
{
    int N,myresult;
    cout<<"Please enter N... ";
    cin>>N;
    myresult=myfunc1(N);
    cout<<myresult;
    return 0;
}
```

C/C++: Write the output MATLAB: Translate into MATLAB
--

Q21

What is the output of the following program? The number N is entered as 5.

```
#include<iostream>
#include<valarray>
using namespace std;
void myfunc2(valarray<int> vec,int num,int& myres)
{ myres=vec[num];}
int main()
{
    int i,t=0,N,myresult;
    valarray<int> v(10);
    for (i=0;i<10;i++) {
        t=t+i;
        v[i]=10*t;}
    cout<<"Please enter N... ";
    cin>>N;
    myfunc2(v,N,myresult);
    cout<<myresult;
    return 0;
}
```

C/C++: Write the output MATLAB: Translate into MATLAB
--

Q22

Let x be a vector with 7 integer elements.

Ask user to enter the elements of the vector x in the **main** program.

Send the vector to a **function** that finds the number of elements which are greater than 3 in numerical value.

Display the result in the **main** program.

In C/C++:

a) Do this question using a void function.

b) Do this question using a non-void function.

Q23

Main program:

- Ask user to enter a real (double) number (x).
- If the number is in the range [-1,1]:
 - ask user to enter an integer number (N),
 - send the numbers (x and N) to the non-void function “func1”,
 - display its output in the main program.
- If the number is not in the range [-1,1]:
 - send the number (x) to the void function “func2”,
 - display its output in the main program.

The non-void function “func1”:

- Input variables: A double number x, an integer number N
- Output variable: A double number y where $y = \sum_{k=0}^N x^k$

The void function “func2”:

- Input variable: A double number x
- Output variable: A double number y where $y = 1/(1-x)$

Q24

The center of mass coordinates of a system of N particles in two dimensions can be given by,

$$x_{cm} = \frac{m_1x_1 + m_2x_2 + m_3x_3 + \cdots + m_Nx_N}{m_1 + m_2 + m_3 + \cdots + m_N}$$

$$y_{cm} = \frac{m_1y_1 + m_2y_2 + m_3y_3 + \cdots + m_Ny_N}{m_1 + m_2 + m_3 + \cdots + m_N}$$

Write a program that calculates the center of mass coordinates in two dimensions.

In the *main program*:

- The number of particles is **N=4**.
- The mass values should be taken from the user and put in an array called **massvec**.
- The x-coordinate values should be taken from the user and put in an array called **xvec**.
- The y-coordinate values should be taken from the user and put in an array called **yvec**.
- Call a *function* to calculate the center of mass coordinates.
- Print the results on the screen.

In the *function*:

- The inputs are **massvec**, **xvec** and **yvec** arrays. (**N** is not an input!)
- The outputs are the center of mass coordinates (**xcm** and **ycm**).

Q25

- Create two vectors **v1** and **v2** with 250 elements.

Set $v1[i]=i$ and $v2[i]=2*i$ where $i = 0 \dots 249$.

- Check if the sum of the elements of **v1** is an **even number**:

- If it is an even number:

- Send **v1** and **v2** to the **function** “vecelemmul” (**not** a void function) to multiply v1 and v2 elementwise (output name: vecmulel).
- Your function should work for **any** two input vectors of equal size.
- Do **not** check if the vectors are of the same size (trust the user).
- Vector length is **not** an input!
- **Display the result in the main program.**

- If it is not an even number:

- Send v1 and v2 to the **void function** “vecavers” to find the average of the elements in v1 (output name:v1ave) and the average of the elements in v2 (output name:v2ave).
- Your void function should work for **any** two input vectors of any size.
- Vector length is **not** an input!
- **Display the results in the main program.**

Q26

Main program:

- Get the number "**x**" from the user.
- Get the number "**N**" from the user.
- Send "**x**" and "**N**" to the function "**myownsinus**" and get the output as "**mysinusvalue**".
- Display "**mysinusvalue**" and the value of "**sin(x)-mysinusvalue**" on the screen.

Function "myownsinus":

- INPUTS: **x**, **N**
- OUTPUT: "**myvalue**", where:

$$\text{myvalue} = \sum_{k=0}^N \left\{ \frac{(-1)^k}{(2k+1)!} x^{(2k+1)} \right\}$$

use the function "**myfactorial**" where needed.


Function "myfactorial":

- INPUT: **number**
- OUTPUT: **Factorial** of the input **number**

Q27

Define the 30x30 matrix **M** with integer elements where $M[i,j]=i+j$. (Here, i is the row number and j is the column number.)

- Send the matrix **M** to the **void** function **mysum** and get the result vector **vec**.
- (i)th element of the result vector **vec** is the sum of the (i)th row of the matrix **M**. (This step will be calculated in the void function **mysum**.)
- Write the result vector **vec** in the file "**myresult.txt**" in the main program.



Read the data from the file **myresult.txt** that you have created in the previous question. Put the data in the vector **myvec** with 30 elements. Find the **maximum** and **minimum** elements of this vector in a **void** function. Display the results in the **main** program.

Q28

Find the root of the equation,

$$2x^2+9x-17=0$$

between $x=0$ and $x=5$ using the bisection method as described below. Use $\epsilon = 10^{-6}$.

Write,

- One **function** to calculate $2x^2+9x-17$. (INPUT: x)
- One **function** for the bisection method (INPUTS: a,b,epsilon). This function should call the function calculating $2x^2+9x-17$.
- A main program that contains the values a, b, epsilon and uses the bisection function to find the root. Check if there is a root between $x=0$ and $x=5$ (the function changes sign if there is a root) and if there is a root, find it using the parameters with the bisection function. Display the result on the screen only in the main program. If there is not a root, print "No root" on the screen.

[illegible]

The Bisection Method:

Given the interval $[a, b]$, define $c = (a + b)/2$. Then

- if $|f(c)| < \text{epsilon}$ then halt, as we found the root (c is the root),
- if $f(c)$ and $f(a)$ have opposite signs, then a root must lie on $[a, c]$, so assign $b = c$,
- else $f(c)$ and $f(b)$ must have opposite signs, and thus a root must lie on $[c, b]$, so assign $a = c$.

Q29

Calculate the following integral using the trapezoid rule by dividing the total interval into 1000 equal subintervals:

$$\int_0^1 \sin(x)\cos(x) dx$$

Your code should include:

- a function for the integrand ($\sin(x) \cos(x)$) [input: x]
- a function for the trapezoid rule (this will call the integrand function) [inputs: integral limits, number of subintervals]
- a main program that calls the trapezoid function to calculate the integral taking 1000 points between the integral limits. Enter the integral limits in the main program.

The trapezoid rule:

$$\int_a^b f(x)dx \approx \frac{1}{2} h[f(x_0) + 2f(x_1) + 2f(x_2) + \cdots + 2f(x_{N-2}) + 2f(x_{N-1}) + f(x_N)]$$

where,

$$\begin{aligned}x_0 &= a \\x_N &= b \\x_k &= x_0 + kh \\h &= \frac{b-a}{N}\end{aligned}$$

for N equal subintervals.

Exact result: 0.3540367092 (any result starting with 0.35 is acceptable.)

Q30

Let us solve the following differential equation:

$$\frac{dy}{dx} = f(x, y) \quad (11)$$

with the initial value $y(x_0) = y_0$.

x_0 is the initial point and we need the value of the unknown function at x_N . The points are

$$x_0, x_1, x_2, x_3, \dots, x_{k-1}, x_k, x_{k+1}, \dots, x_N \quad (12)$$

Thus we have

$$x_k = x_0 + kh \quad (13)$$

where $h = \frac{x_N - x_0}{N}$ is the spacing between the points and $k = 0, 1, 2, \dots, N$.

The formula for the Euler's method is

$$y_{k+1} = y_k + hf(x_k, y_k) \quad (14)$$

*

$$\frac{dy}{dx} = x + y$$

with the initial value $y(0) = 1$. Find $y(5)$ using 100 steps.

Q31

Second order RK (RK2)

$$y_{k+1} = y_k + \frac{h}{2}[f(x_k, y_k) + f(x_{k+1}, y_k + hf(x_k, y_k))]$$

Fourth order RK (RK4)

$$y_{k+1} = y_k + \frac{h}{6}(F_1 + 2F_2 + 2F_3 + F_4)$$

$$F_1 = f(x_k, y_k)$$

$$F_2 = f\left(x_k + \frac{h}{2}, y_k + \frac{h}{2}F_1\right)$$

$$F_3 = f\left(x_k + \frac{h}{2}, y_k + \frac{h}{2}F_2\right)$$

$$F_4 = f\left(x_k + h, y_k + hF_3\right)$$

Do the previous question using RK2 and RK4 with 100 steps.

Exact result:
y(5)=290.8263182

Q32

Bubblesort algorithm: A simple sorting algorithm that works by repeatedly stepping through the list to be sorted, comparing each pair of adjacent items and swapping them if they are in the wrong order. The pass through the list is repeated until no swaps are needed, which indicates that the list is sorted. The algorithm gets its name from the way smaller elements "bubble" to the top of the list.
[Wikipedia]

Step-by-step example:

Let us take the array of numbers "5 1 4 2 8", and sort the array from lowest number to greatest number using bubble sort. In each step, elements written in **bold** are being compared.

First Pass:

(**5** 1 4 2 8) \rightarrow (**1** **5** 4 2 8), Here, compares the first two elements, and swaps since $5 > 1$.

(1 **5** 4 2 8) \rightarrow (1 **4** **5** 2 8), Swap since $5 > 4$

(1 4 **5** 2 8) \rightarrow (1 4 **2** **5** 8), Swap since $5 > 2$

(1 4 2 **5** 8) \rightarrow (1 4 2 **5** 8), already in order ($8 > 5$), algorithm does not swap them.

Second Pass:

(**1** **4** 2 5 8) \rightarrow (**1** **4** 2 5 8)

(1 **4** 2 5 8) \rightarrow (1 **2** **4** 5 8), Swap since $4 > 2$

(1 2 **4** 5 8) \rightarrow (1 2 **4** 5 8)

(1 2 4 **5** 8) \rightarrow (1 2 4 **5** 8)

Now, the array is already sorted, but our algorithm does not know if it is completed. The algorithm needs one **whole** pass without **any** swap to know it is sorted.

Third Pass:

(**1** **2** 4 5 8) \rightarrow (**1** **2** 4 5 8)

(1 **2** 4 5 8) \rightarrow (1 **2** 4 5 8)

(1 2 **4** 5 8) \rightarrow (1 2 **4** 5 8)

(1 2 4 **5** 8) \rightarrow (1 2 4 **5** 8)

No swapping in the third pass, so the array is sorted.

You will write a **main** program and a function "**bubblesort**":

- Enter a **vector** to be sorted in the main program.
 - Send the vector to the function "**bubblesort**" and print out the resulting (sorted) vector in the main program. (Your function should work for vectors in **any** length.)
-

Q33

Do the following exercises in SageMath:

- 1) Define a polynomial depending on x and take the first derivative with respect to x .
- 2) Define a polynomial depending on x and take the third derivative with respect to x .
- 3) Write an indefinite integral and take it.
- 4) Write a definite integral and take it.
- 5) Write a function of x and take its limit as x goes to zero.
- 6) Write a first order differential equation without an initial/boundary condition and solve it.
- 7) Write a first order differential equation with an initial/boundary condition and solve it.
- 8) Write a second order differential equation without initial/boundary conditions and solve it.
- 9) Write a second order differential equation with initial/boundary conditions and solve it.
- 10) Simplify $(1/(x+1))+(x/(x-6))-((5*x-2)/(x^2-5*x-6))$ to see it is $(x-4)/(x-6)$.
- 11) Show $\sin(x)$ and $\cos(x)$ plots ($x=0..3*\pi$) on the same graph.
- 12) Plot $\sin(x*y)+\cos(x*y)$ ($x=-\pi..\pi$), ($y=-\pi..\pi$) as a three dimensional graph.

Q34

Plot the geometric series for various upper summation limits instead of infinity to see the convergence (use $a=1$, $r=0.5$):

$$\textit{Geometric Series: } \sum_{n=0}^{\infty} ar^n = a + ar + ar^2 + \dots$$

Geometric Series converges to sum $\frac{a}{1-r}$ iff $|r| < 1$

Q35

Plot the following functions along with their series expansions to find the number of terms needed for an error $< 10^{-6}$

(error = |function value - series value|)

Use $x=0.1$, $x=1$, $x=10$

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} \dots$$

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} \dots$$

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} \dots$$

Q36

In MATLAB/GNU Octave, we can produce a random real number between $[0,1]$ using the command `rand()`.

A drunken man wants to go home. Now, he is in front of the bar and his home is at the opposite side of a circular street.

Suppose he is standing at the 50th meter of the street and you can denote the location of his home as at the 100th or the 0th meter of the street.

He is so drunken that he steps forward with a possibility of 50% and another 50% backwards. The step-size of the man is 1 meter.

Write a program to find how many steps should the drunken man walk to get his home.

Plot his position versus step number (i^{th} position - i^{th} step).

(Alcohol is not your friend!)