

Assignment 8

Write octave programs to solve below questions. Try to understand questions by yourself rather than asking TAs or me for clarifications. This habit will be useful in examinations both lab and written when I or TAs wont be solving any queries regarding questions asked in exams.

1. Create a program that gives nth prime number starting from 2. That is if user gives input 3 it should print third primer number in prime number series, that is 5.
2. Create a program that takes keeps taking numbers from user. When user enters number 0 the program stops asking for more numbers and prints maximum and minimum numbers among numbers entered by user so far.
3. Create a program that first asks user about dimensions of matrix. Then asks users to enter values to fill all elements of two matrix of dimensions entered by user. Then display sum of two matrices. For example

```
Enter number of rows: 2
Enter number of columns: 2
Enter matrix 1 element (1,1): 1
Enter matrix 1 element (1,2): 2
Enter matrix 1 element (2,1): 3
Enter matrix 1 element (2,2): 4
Enter matrix 2 element (1,1): 5
Enter matrix 2 element (1,2): 6
Enter matrix 2 element (2,1): 7
Enter matrix 2 element (2,2): 8
Sum is:ans =
```

```
6 8
10 12
```

4. Create a program that generates following combination of Fibonacci series and prime numbers. In the below sequence numbers at odd position (1, 3, 5, etc.) are of Fibonacci series and other numbers are of prime number series. Example sequence is 1, 2, 1, 3, 2, 5, 3, 7. You must take input from user on how many numbers of this combined sequence to generate.

5. Create a program that takes coefficients and constants of linear equation in two variables from user and prints values of x and y. (That is, assuming equations are $a_1x+b_1y=c_1$ and $a_2x+b_2y=c_2$, take values of a_1 , b_1 , c_1 , a_2 , b_2 and c_2 from user and print values of x and y that solve the equation.) You can assume the equation entered by user has unique solution and equations do not represent parallel or overlapping lines).
6. If we try to generate graph of $\tan(x)$ using following commands

```
x=linspace(-10,10,10000);  
y=tan(x);  
plot(x,y);
```

then the plot that we see is not similar to what we usually study as graph for $\tan(x)$. Generate graph as shown in figure(1) below for $\tan(x)$. (Hint: You might require use of function 'axis') Store this graph in file

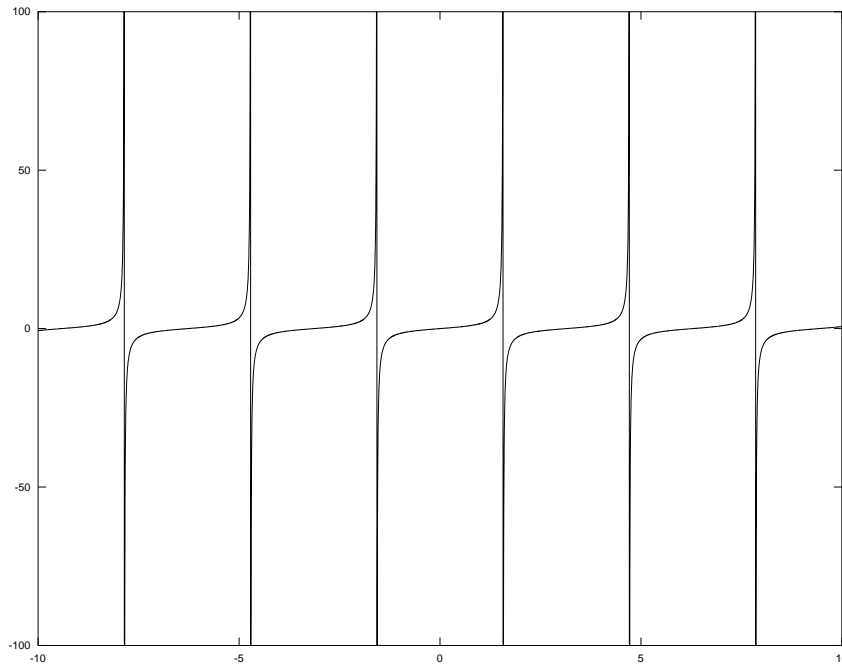


Figure 1: Graph of $\tan(x)$

tanx.png using your octave program.

7. Write a program that takes a number from user and prints 'It is palindrome' if number is palindrome, else prints 'It is not palindrome'. The program must take input as number and not as string.
8. Write a program that takes string from user and prints 'It is palindrome' if it is palindrome, else prints 'It is not palindrome'. The program must take input as string.
9. Write a program that takes unlimited numbers from user. The program stops asking for more numbers when user enters number '0'. Then program should sort all numbers given so far and print them in ascending order. The program should support negative real numbers.
10. Write a program that takes series of strings from user. The program stops asking for more strings when user enters 'exit'. Then program should sort all strings given so far and print them in ascending order. For purpose of comparison assume 'a' < 'b' < 'c' and so on.
11. Write a program that encrypts message given by user. For encryption the program should convert all letters 'a' to 'b', all letters 'b' to 'c', all 'c' to 'd' and so on. It should convert all letters 'z' to 'a'. Example if user enters 'hello' as input the output of program should be 'ifmmp'. Only capital and small letters should be rotated. Other characters like '+', '-', '@', ' ' etc. should not be changed.
12. Write a program that decrypts message encrypted using before program. For example if user enters message 'ifmmp' then program should output 'hello'.
13. Write a program that encrypts message using similar algorithm as used in functions above. But rather than just advancing the characters by 1, the program should ask users how many characters advancement should be used for encryption. For example if user enters string 'hello' and number 0, the output should be 'hello'. If user enters string 'hello' and number 1 the output should be 'ifmmp'. If user enters string 'hello' and number 2 the output should be 'jgnnq'. In this program also only capital and small letters should be rotated. Other characters like '+', '-', '@', ' ' etc. should not be changed. The program should also accept inputs greater than or equal to 26. For 26 the output would be same as that for 0, for 27 the output would be same as that of 1 and so on. (This is known as Caesar cipher, specifically when we shift characters by 3).

14. Write a program that decrypts message encrypted using above program. Hence if user enters string 'jgnnq' and that number 2 was used during encryption, then program should print original message was 'hello' before encryption.
15. Write a program that takes set of original characters and then set of transformed characters. For example

Original : ABCDEF
Transformed: ZAJWBR

Then when user enters message as 'BED' the encrypted/transformed message should be 'ABW'. To decrypt the message the user will run the program again and this time he would enter

Original : ZAJWBR
Transformed: ABCDEF

and then when he enters text 'ABW' the original message 'BED' should be printed. (This is known as substitution cypher.) You can assume that characters would not repeat in original string, that is you will not get original string as 'aabcde'. You can also assume that characters will not repeat in transformed string, that is you wont get transformed string like 'basefe'. You are supposed to change only those characters in message for which the transformation is defined, that is if original, transformed and message string are as shown below then

Original : ZAJWBR
Transformed: ABCDEF
Message : HELLO WORLD

output should be 'HELLO DOFLD'. You can convert all original, transformed and message strings to upper case before encryption / decryption.

16. Write a program that allows user to do basic mathematical operations ('+', '-', '*' and '/') on real numbers. The program should print a menu and ask user which operation user wants to perform. Then program should ask user to enter two numbers and then print result of operation on those two number. The user should again get choice on which operation to perform. The program should continue until user chooses to exit. Sample menu printed by program can be like

Choose operation to perform:

1. Add numbers
2. Subtract numbers
3. Multiply numbers
4. Divide numbers
5. End program

Enter your choice :

17. Write a program that allows user to manage array of numbers. The program should print following menu

- (a) Print numbers in current array
- (b) Initialize array by new set of items
- (c) Delete all elements from array
- (d) Delete element at n^{th} position of array
- (e) Insert element at n^{th} position in array.
- (f) End program

and let user choose what operation should be done. The program should keep printing this menu after every operation until user chooses to exit.

18. Write a program that allows user to manage array of strings. The program should print following menu

- (a) Print strings in current array
- (b) Initialize array by new set of strings
- (c) Delete all strings from array
- (d) Delete string at n^{th} position of array
- (e) Insert string at n^{th} position in array.
- (f) End program

and let user choose what operation should be done. The program should keep printing this menu after every operation until user chooses to exit.

19. Write a program that asks user to enter number and based on following logic tells whether number is leap year or not. A year is a leap year if it is divisible by 400; OR if it is not divisible by 100 and divisible by 4. That is to say,

Year 5 is not a leap year
Year 2000 is a leap year
Year 1900 is not a leap year
Year 1800 is not a leap year
Year 1600 is a leap year
Year 2004 is a yeap year.

20. Write a program that asks users number of seconds that have passed since 1 Januray, 1970 and then print current date and time. The program should account for leap year properly as per definition of leap year mentioned in above problem.