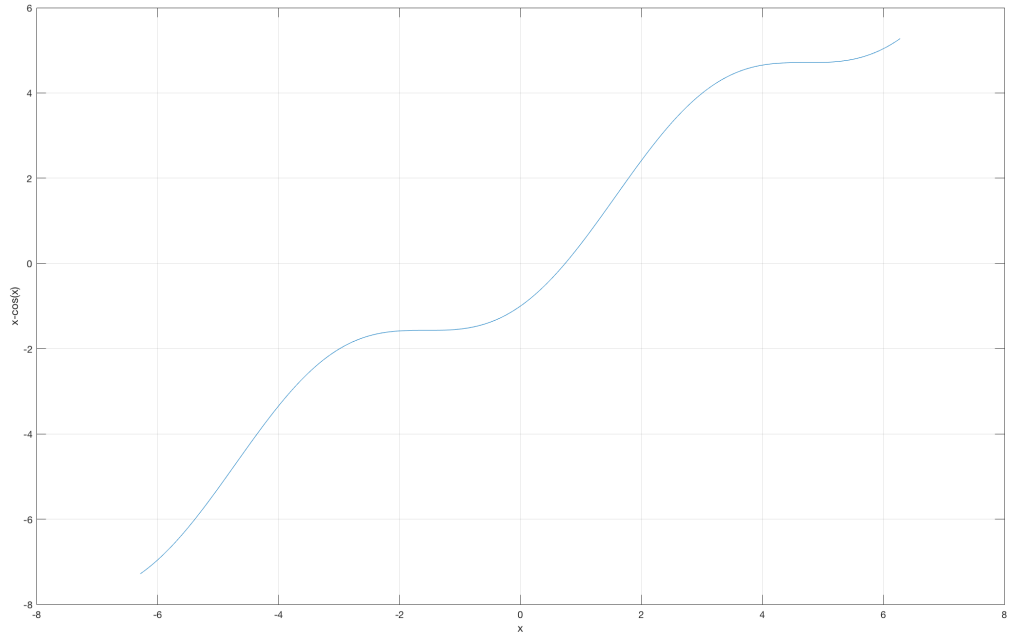
ENSC 180 - Assignment 2

Shengcong Zhou
Lab group 9
sza111@sfu.ca

1.

cos(x)=x

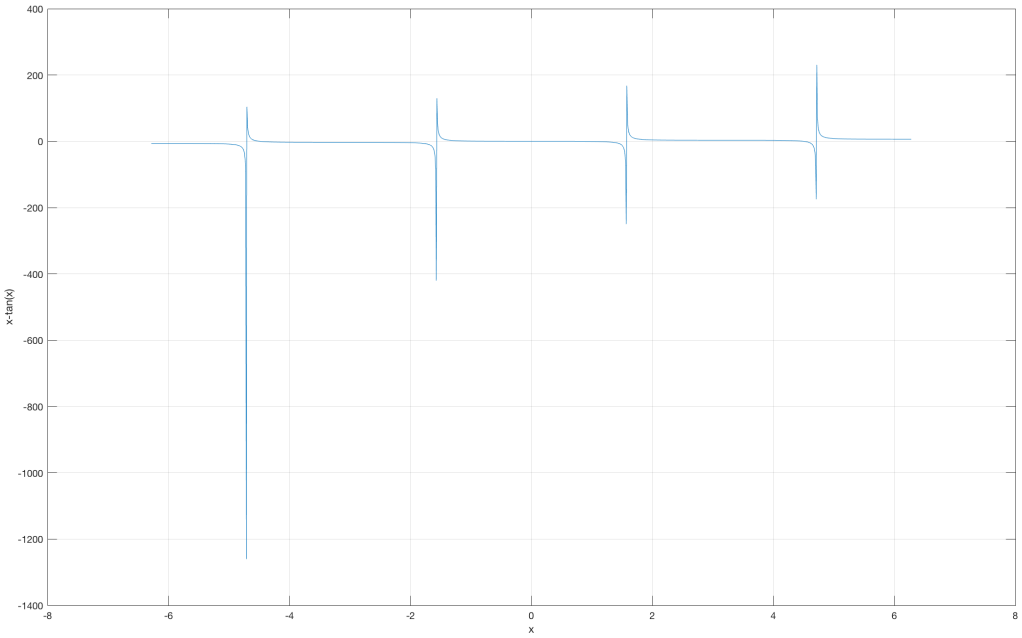
```
x = -2*pi:0.01:2*pi;
y = x - cos(x);
plot(x,y)
grid
xlabel('x')
ylabel('x-cos(x)')
for i=1:1256
    %Product of y(i)*y(i+1)=negative when graph crosses x-axis
    if y(i)*y(i+1) < 0
        disp(x(i))
    end
end
```



The root found by the program is 0.73681

tan(x)=x

```
x = -2*pi:0.01:2*pi;
y = x - tan(x);
plot(x,y)
grid
xlabel('x')
ylabel('x-tan(x)')
for i=1:1256
    if y(i)<1 && y(i+1) <1 %Prevent asymptotes to be counted as roots
        %Product of y(i)*y(i+1)=negative when graph crosses x-axis
        if y(i)*y(i+1) < 0
            disp(x(i))
        end
    end
end
```



The roots found by the program are: -4.5032, -0.0031853, 4.4868

2.

```
x=[73,92,65,41,37,80,67,54,90,82,85,69,76,74,82,87,69,78,85];
fprintf('Mark\tGrade\n');
for i=1:19
   if x(i) >= 90
      fprintf('%d\tA+\n',x(i));
   elseif x(i)>=80
      fprintf('%d\tA\n',x(i));
   elseif x(i)>=75
      fprintf('%d\tB+\n',x(i));
   elseif x(i)>=68
      fprintf('%d\tB\n',x(i));
   elseif x(i)>=60
      fprintf('%d\tC+\n',x(i));
   elseif x(i)>=50
      fprintf('%d\tC\n',x(i));
   elseif x(i)>=40
      fprintf('%d\tD\n',x(i));
   else
      fprintf('%d\tF\n',x(i));
   end
end
```

Output of the program

```
Mark   Grade
73     B
92     A+
65     C+
41     D
37     F
80     A
67     C+
54     C
90     A+
82     A
85     A
69     B
76     B+
74     B
82     A
87     A
69     B
78     B+
85     A
```

I used fprintf for this version of the program however I also have a version where I use a string array for the grade down below. I do prefer fprintf version because it doesn't print quotation marks.

```
x=[73,92,65,41,37,80,67,54,90,82,85,69,76,74,82,87,69,78,85];
grade=strings([1,19]); %Initialize grade as an empty string array
disp('    Mark    Grade')
for i=1:19
    %Assign a letter grade for the corresponding grade
    if x(i) >= 90
       grade(i)= 'A+';
    elseif x(i)>=80
       grade(i)= 'A';
    elseif x(i)>=75
       grade(i)= 'B+';
    elseif x(i)>=68
       grade(i)= 'B';
    elseif x(i)>=60
       grade(i)= 'C+';
    elseif x(i)>=50
       grade(i)= 'C';
    elseif x(i)>=40
       grade(i)= 'D';
    else
       grade(i)= 'F';
    end
end
disp([x' grade'])
```

Output of the program

```
    Mark    Grade
    "73"    "B"
    "92"    "A+"
    "65"    "C+"
    "41"    "D"
    "37"    "F"
    "80"    "A"
    "67"    "C+"
    "54"    "C"
    "90"    "A+"
    "82"    "A"
    "85"    "A"
    "69"    "B"
    "76"    "B+"
    "74"    "B"
    "82"    "A"
    "87"    "A"
    "69"    "B"
    "78"    "B+"
    "85"    "A"
```

3.

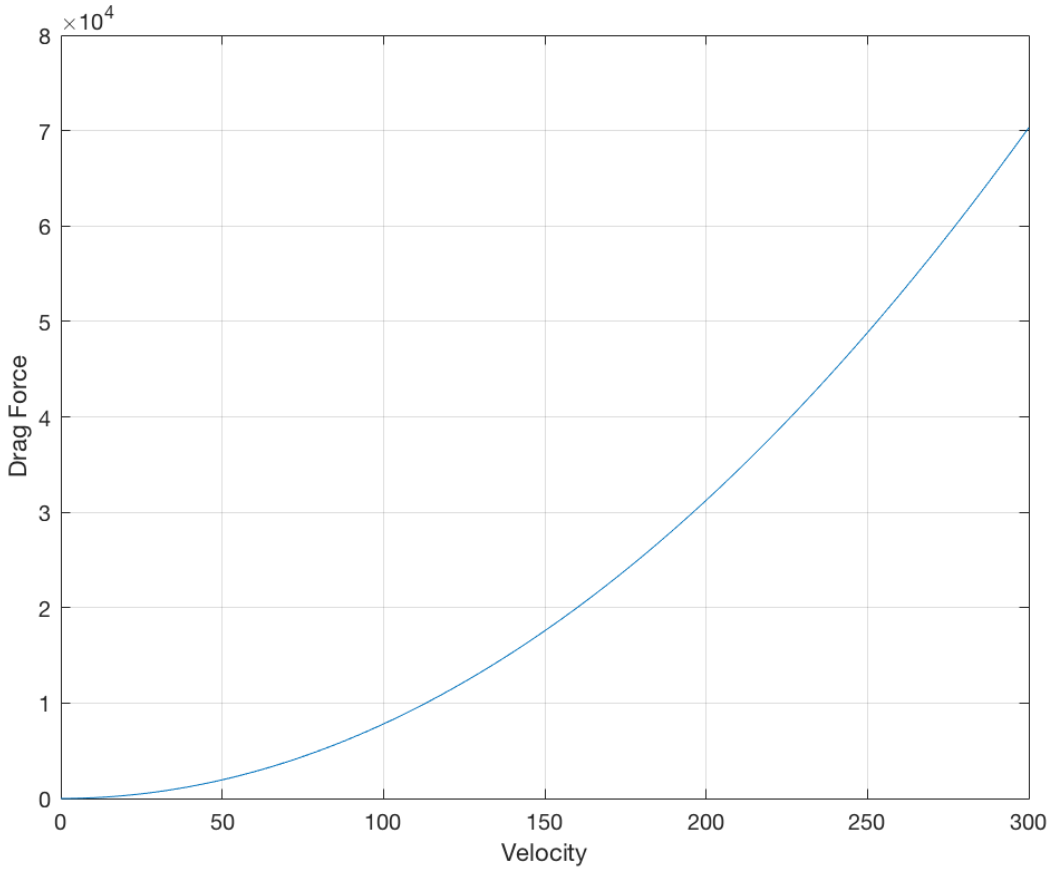
```
Fd=input('Enter the drag force\n');
p=input('Enter the air density\n');
V=input('Enter the velocity\n');
A=input('Enter the surface area\n');

Cd=Fd/(p*V^2*A/2); %Calculate Cd from inputed data

V=0:1:300;
Fd=Cd.*p.*(V.^2).*A./2;
plot(V,Fd)
grid
xlabel('Velocity')
ylabel('Drag Force')
```

Output of the program when Drag force= 20000N; air density = 1*10E-06 kg/m3; V=160 km/h; and area=0.9 m2

```
Enter the drag force
20000
Enter the air density
1*10E-6
Enter the velocity
160
Enter the surface area
0.9
Drag coefficient=1.736111e+05
```



4.

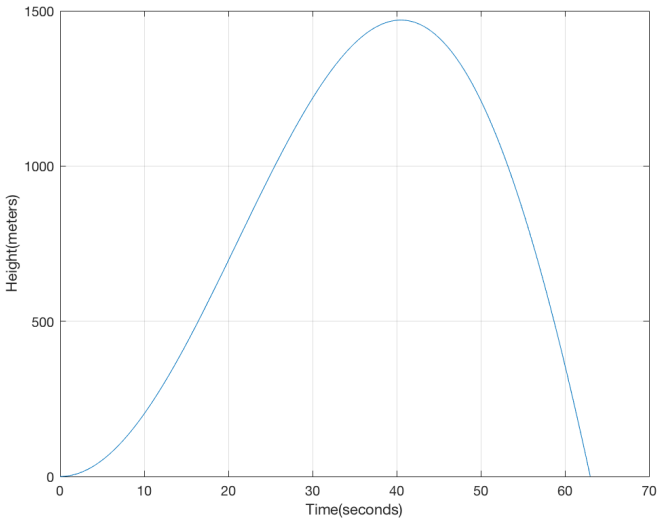
```
t = 0:0.01:70; %Choose a random range of t to see the tendency of polynomial
H = 2.13.*t.^2 - 0.0013.*t.^4 + 0.000034.*t.^4.751;
MaxHeight=max(H);
fprintf('Max Height = %f\n',MaxHeight)
for i=2:7000
   %Height is at max when the heights right before and after are lower
   if  H(i+1)<H(i) && H(i-1)<H(i)
      fprintf('Time to reach max height= %f\n',t(i))
      TimeMaxHeight=t(i);
   %Producet of H(i)*H(i+1)=negative when graph crosses x-axis
   elseif H(i)*H(i+1) < 0
      fprintf('Time landing= %f\n',t(i))
      TimeLanding=t(i);
   end
end

clear t
clear H

t=0:0.01:TimeLanding;
H = 2.13.*t.^2 - 0.0013.*t.^4 + 0.000034.*t.^4.751;
plot(t,H)
grid
xlabel('Time(seconds)')
ylabel('Height(meters)')
```

Output of the program

```
Max Height = 1470.187263
Time to reach max height= 40.500000
Time landing= 63.010000
```



For 0<=t<=70
H'(t)=0 at t=0 and t=40.5012
The later is the time for the rocket to reach the maximum height.

H(40.5012)=1470.19
The Height at 40.5012, which is the local max of polynomial H(t).

H(t)=0 at t=0 and t=63.014
The roots of H(t) and the later is the landing time.

Comparing the values to the output of the MATLAB program, answers obtained seem reasonable.

5.

```
T1=[4.0,1.5,6.0,0.75,12.0,72.0,0.0,0.0,4.0,2.75];
T2=[48.0,0.0,5.5,1.00,3.00,2.00,1.5,2.5,4.0,1.5];
T3=[1.0,1.5,5.0,8.0,1.5,2.0,1.5,1.75,12.0,2.0];
T=[60*T1;60*T2;60*T3]; %Convert hours to minutes
Cost=zeros(1,3); %Initialize cost

for person=1:3
   fprintf('Person%d\n',person)
   for day=1:10
      fprintf('Day%d, ',day)
      Minutes=(mod(T(person,day),1440));  %Minutes left after subtracting the time counted as
a day(Days)
      Days=(T(person,day)-Minutes)/1440; %# of days in the hours
      if Minutes>300 %If minutes >300 long term daily max is the cheapest
         Cost(person)=Cost(person)+(Days*18)+18;
         fprintf('cost=%d, LT max\n',Cost(person));
      elseif Minutes==0 %If minutes is 0 there is no charge except daily charges
         Cost(person)=Cost(person)+(Days*18);
         fprintf('cost=%d, No minute wise charge\n',Cost(person));
      elseif Minutes<=135 %If minutes is <=135 short term is cheaper than long term
         if Minutes<=30 %If minutes is <=30 no additional charge
            Cost(person)=Cost(person)+(Days*18)+2.5;
            fprintf('cost=%d, ST min\n',Cost(person));
         elseif mod((Minutes-30),15)==0
            Charges=(Minutes-30)/15;
            Cost(person)=Cost(person)+(Days*18)+2.5+(Charges*1);
            fprintf('cost=%d, ST\n',Cost(person));
         else
            Charges=(Minutes-30-mod((Minutes-30),15))/15+1;
            Cost(person)=Cost(person)+(Days*18)+2.5+(Charges*1);
            fprintf('cost=%d, ST\n',Cost(person));
         end
      else %If 135<minutes<=300 long term is cheaper than short term
         if Minutes<=180 %If 135<minutes<=180 no additional charge
            Cost(person)=Cost(person)+(Days*18)+10;
            fprintf('cost=%d, LT min\n',Cost(person));
         elseif mod((Minutes-180),60)==0
            Charges=(Minutes-180)/60;
            Cost(person)=Cost(person)+(Days*18)+10+(Charges*3);
            fprintf('cost=%d, LT \n',Cost(person));
         else
            Chrges=(Minutes-180-mod((Minutes-180),60))/60+1;
            Cost(person)=Cost(person)+(Days*18)+10+(Charges*3);
            fprintf('cost=%d, LT \n',Cost(person));
         end
      end
      if mod(day,7)==0 && Cost(person)>80 %If the sum over 7 days>80 apply weekly max
         Cost(person)=80;
         fprintf('weekly max applied\n');
      end
   end
end
```

```
fprintf('\nThe minimum costs are\nPerson 1= %d\nPerson 2= %d\nPerson 3=
%d\n',Cost(1),Cost(2),Cost(3))
```

Output of the program

```
Person1
Day1, cost=13, LT
Day2, cost=1.950000e+01, ST
Day3, cost=3.750000e+01, LT max
Day4, cost=41, ST
Day5, cost=59, LT max
Day6, cost=113, No minute wise charge
Day7, cost=113, No minute wise charge
weekly max applied
Day8, cost=80, No minute wise charge
Day9, cost=93, LT
Day10, cost=103, LT min
Person2
Day1, cost=36, No minute wise charge
Day2, cost=36, No minute wise charge
Day3, cost=54, LT max
Day4, cost=5.850000e+01, ST
Day5, cost=6.850000e+01, LT min
Day6, cost=77, ST
Day7, cost=8.350000e+01, ST
weekly max applied
Day8, cost=90, LT min
Day9, cost=103, LT
Day10, cost=1.095000e+02, ST
Person3
Day1, cost=4.500000e+00, ST
Day2, cost=11, ST
Day3, cost=27, LT
Day4, cost=45, LT max
Day5, cost=5.150000e+01, ST
Day6, cost=60, ST
Day7, cost=6.650000e+01, ST
Day8, cost=74, ST
Day9, cost=92, LT max
Day10, cost=1.005000e+02, ST

The minimum costs are
Person 1= 103
Person 2= 1.095000e+02
Person 3= 1.005000e+02
```

I assumed a week is counted from day 1 and when parking hours exceed 24hrs it counts days and minutes. e.g.(72.5hrs=3days+30min so it will charge daily max 3 times and calculate the minimum cost for 30 minutes and the charge is the sum)