2016 CS333000

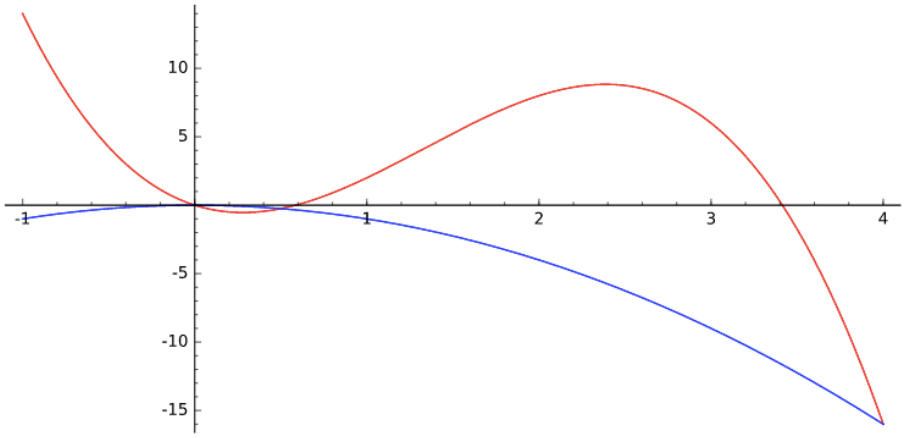
S3 (HW5) Ans

Q1.

```
f = -2*x*(x**2 - 4*x + 2)

g = -x**2

solve(f==g,x)

plot(f, -1, 4, color="red")+plot(g, -1, 4, color="blue");

print "\n"

print "integrate : ", integrate(g-f, x, 0, 0.5)+integrate(f-g, x, 0.5, 4);
```

result:

```
[x == 4, x == (1/2), x == 0]
```




```
integrate :  32.3125
```

Q2.

f1(x) = sin(x)/x;

f2(x) = 0;

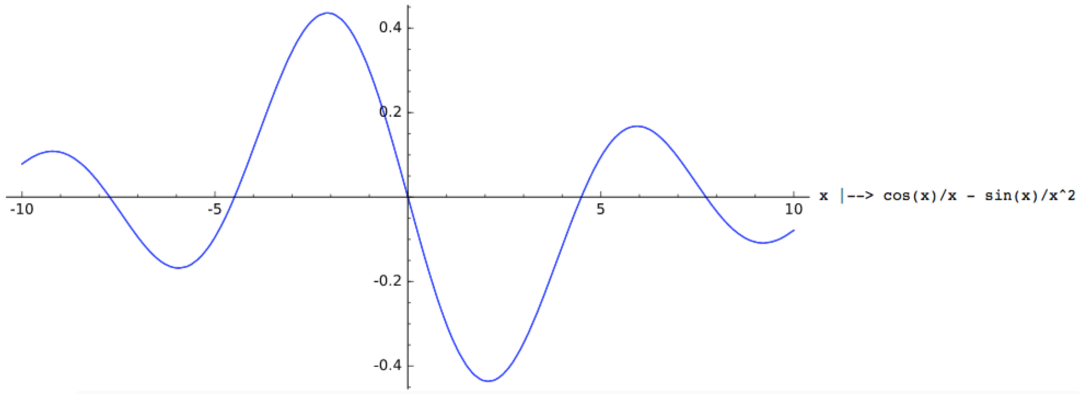
g = Piecewise([[(-infinity,0), f1], [(0, infinity), f1]]); g;

y = derivative(g); y

plot(derivative(f1), -10, 10);derivative(f1)

result:

```
Piecewise defined function with 2 parts, [[(-Infinity, 0), x |--> sin(x)/x], [(0, +Infinity), x |--> sin(x)/x]]
Piecewise defined function with 2 parts, [[(-Infinity, 0), x |--> cos(x)/x - sin(x)/x^2], [(0, +Infinity), x |--> cos(x)/x - sin(x)/x^2]]
```



Q3.

```
def MidpointRule(myf,a,b,n):
    deltax = (b-a)/n;
    ans = 0;
    for i in range(0,n):
        x_star = ((a+deltax*i)+(a+deltax*(i+1)))*0.5;
        ans += myf(x_star);
    return deltax*ans;

def SimpsonsRule(myf,a,b,n):
# approximation way, only works when n is a even number
    if n%2 == 1:
        n += 1;
    deltax = (b-a)/n
    multi = [4,2]*int(n/2)
    multi = [1] + multi[:n-1] + [1]
    ans = 0
    for i in range(0, n+1):
        ans += myf(a+deltax*i)*multi[i];
    return (deltax/3)*(ans);
```

```
f = -2*x*(x**2 - 4*x + 2)
correct = integrate(f, x, 0.5, 4); N(correct);
mid = MidpointRule(f, 0.5, 4, 12); N(mid);
sim = SimpsonsRule(f, 0.5, 4, 12); N(sim);
print "mid point error", N(abs(correct - mid));
print "sim point error", N(abs(correct - sim));
print "mid point accuracy", N(100- 100*abs(correct - mid)/correct );
print "sim point accuracy", N(100- 100*abs(correct - sim)/correct);
```

result:

```
10.8645833333333
11.0010489004630
10.8645833333333
mid point error 0.136465567129630
sim point error 1.77635683940025e-15
mid point accuracy 98.7439410887397
sim point accuracy 100.000000000000
```