MATH 156 LAB 5

BYUNG DO PARK

We introduce two new Riemann sums to approximate integrals. The Trapezoid Rule
TRAP(n)=(LHS(n)+RHS(n))/2

and the Midpoint Rule, which instead of computing using the values of $f(x)$ at the left endpoint $x\_(i-1)$ and the right endpoint x_i, it uses the midpoint $\frac{x\_(i-1)+x\_i}{2}$. So we have

$$\text{MID}(n)=\sum_{i=1}^{n} f\left(\frac{x\_(i-1)+x\_i}{2}\right).$$

Maple has commands that will plot for us the midpoint rule and compute the midpoint rule.

We should introduce the student package. Let us introduce the function $f(x)=\sqrt{x}$ and consider the integral $\int_1^4 \sqrt{x}\, dx$.

```
> f:=x->sqrt(x);
```

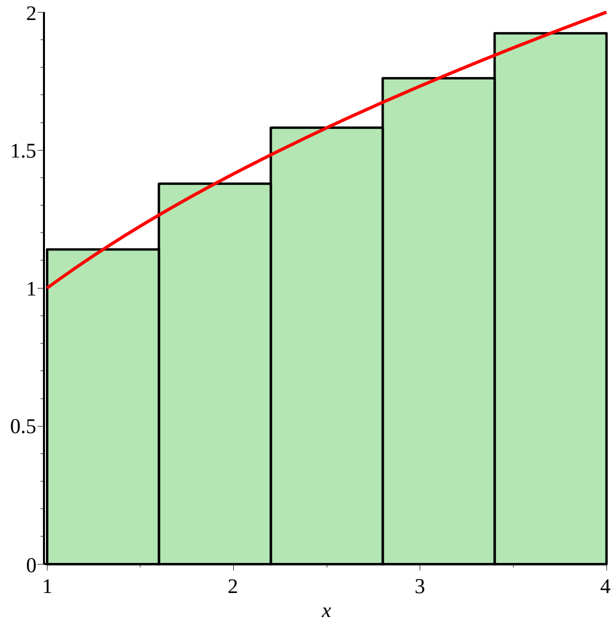
$$f := x \rightarrow \sqrt{x}$$

```
> with(student):
```

Topic 1: Midpoint Rule

The command for graphing the midpoint rule is middlebox(function(x), x=lowerlimit..upperlimit, number of subintervals);

```
> middlebox(f(x), x=1..4, 5);
```



The command to compute numerically the midpoint rule is middlesum(function (x), x=lowerlimit..upperlimit, number of subintervals);

```
> middlesum(f(x), x=1..4, 5);
```

$$\frac{3}{5} \sum_{i=0}^{4} \sqrt{\frac{13}{10} + \frac{3}{5} i}$$

As you see, Maple does not evaluate it immediately, so we use the evalf command.

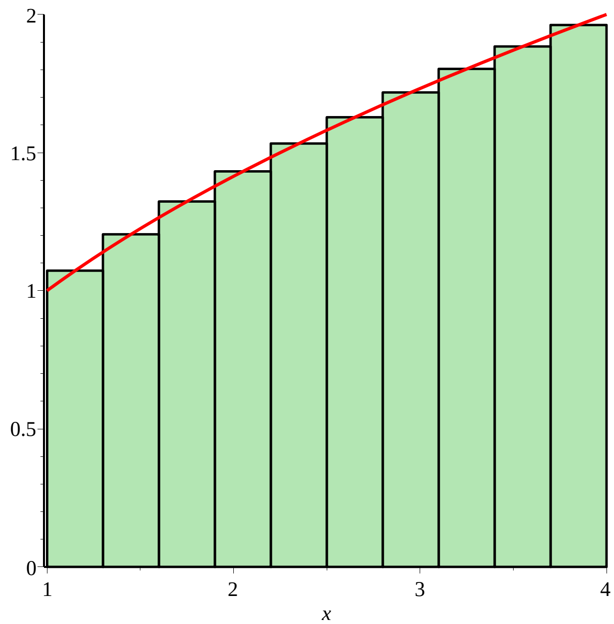
```
> evalf(middlesum(f(x), x=1..4, 5));
```

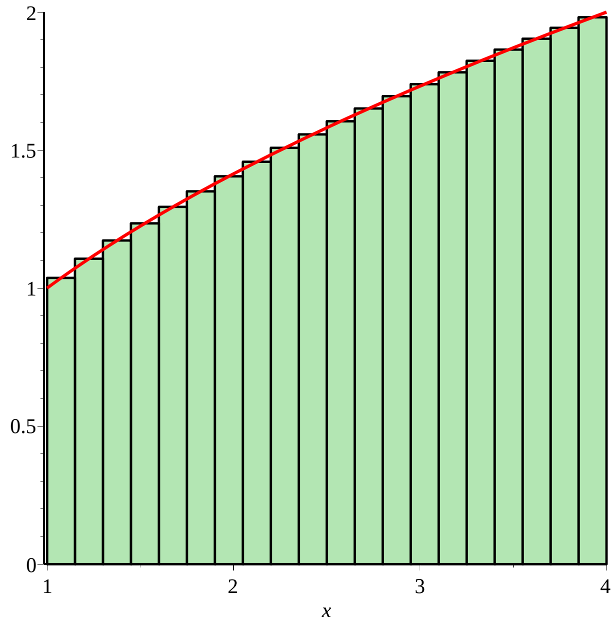
$$4.670363534$$

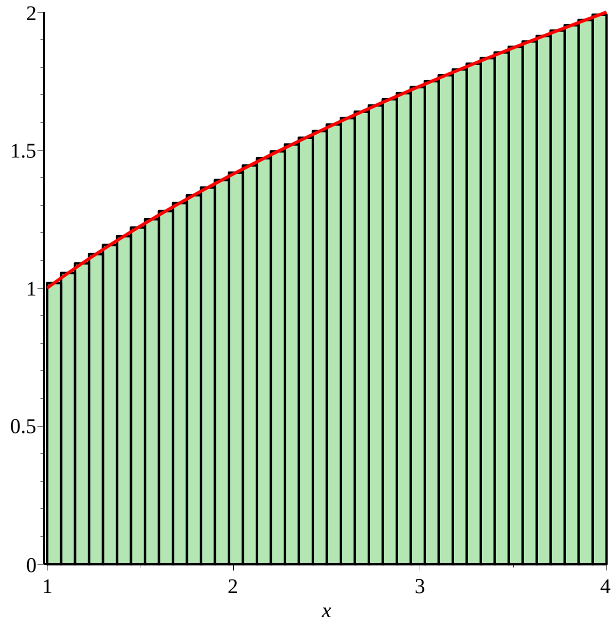
Write commands that show on the graph the midpoint rule with 10, 20, 40 subintervals. Write commands that name these graphs

and the graph with 5 subintervals above. Write commands that evaluate the midpoint rule with 10, 20, 40 subintervals.

```
> middlebox(f(x), x=1..4, 10); middlebox(f(x), x=1..4, 20);
  middlebox(f(x), x=1..4,40);
```




2
1.5
1
0.5
0
1
2
3
4
x




```
> msum10 := middlebox(f(x), x = 1..4, 10) : msum20 := middlebox(f(x), x = 1..4, 20) :
      msum40 := middlebox(f(x), x = 1..4, 40) : msum5 := middlebox(f(x), x = 1..4, 5) :
```

```
> evalf(middlesum(f(x), x = 1..4, 10)); evalf(middlesum(f(x), x = 1..4, 20));
      evalf(middlesum(f(x), x = 1..4, 40));
```

4.667600664

4.666900820

4.666725247 **(1)**

Topic 2: Comparing the midpoint rule with the left-hand sums and right-hand sums.

Write commands that name the graphs of the left-hand sums and right-hand sums with 5, 10, 20, 40 subintervals.

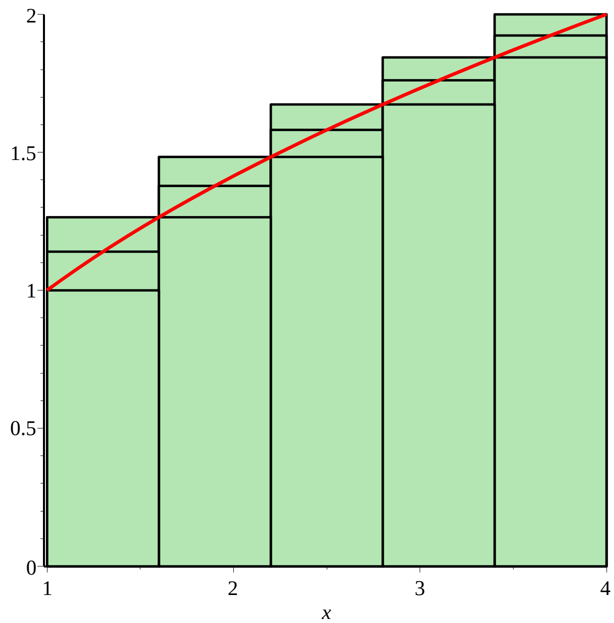
```
> lhs10 := leftbox(f(x), x = 1..4, 10) : lhs20 := leftbox(f(x), x = 1..4, 20) : lhs40
```

≔ **leftbox**(f(x), x = 1 ..4, 40) : lhs5 ≔ **leftbox**(f(x), x = 1 ..4, 5) :

```
> rhs10 := rightbox(f(x), x = 1 ..4, 10) : rhs20 := rightbox(f(x), x = 1 ..4, 20) : rhs40
      := rightbox(f(x), x = 1 ..4, 40) : rhs5 := rightbox(f(x), x = 1 ..4, 5) :
```

Write commands that show on the same graph the left-hand sums, the right-hand sums and the midpoint sums with the same number of subintervals. Do not forget to introduce the plots package. What do you notice? Which are larger, the left-hand sums, right-hand sums, or midpoint sums? Can you explain it?

```
> with(plots) :
> display(lhs5, msum5, rhs5);
```



Write commands that compute the left-hand sums and right-hand sum and midpoints sums numerically with 5, 10, 20, 40, 80, 160, 320, 640, 1280, 2560 subintervals. You can use a loop. What do you notice?

```
> for k from 0 to 9 do N := 5·2^k; evalf(leftsum(f(x), x = 1 ..4, N)); evalf(middlesum(f(x), x
    = 1 ..4, N)); evalf(rightsum(f(x), x = 1 ..4, N)); od;
```

$$N := 5$$

4.359227824

4.670363534

4.959227824

$$N := 10$$

4.514795679

4.667600664

4.814795679

$$N := 20$$

4.591198172

4.666900820

4.741198172

$$N := 40$$

4.629049495

4.666725247

4.704049495

$$N := 80$$

4.647887370

4.666681312

4.685387370

$$N := 160$$

4.657284343

4.666670329

4.676034343

$$N := 320$$

4.661977336

4.666667582

4.671352336

$$N := 640$$

4.664322459

4.666666896

4.669009959

$N := 1280$

4.665494677

4.666666725

4.667838427

$N := 2560$

4.666080701

4.666666680

4.667252576 **(2)**

>

Topic 3: Trapezoid rule.

It is easy to calculate the trapezoid rule, as it is the average of the left-hand sum and the right-hand-sum.

Write commands that calculate the trapezoid rule, left-hand sums and right-hand sums with 5, 10, 20, 40, 80, 160, 320, 640, 1280, 2560 subintervals. You can use a loop. What do you notice? Which are larger, smaller? Why?

```
> for k from 0 to 9 do N := 5·2^k; evalf(leftsum(f(x), x = 1 ..4, N)); evalf(0.5·((leftsum(f(x),
    x = 1 ..4, N)) + (rightsum(f(x), x = 1 ..4, N)))); evalf(rightsum(f(x), x = 1 ..4, N)); od;
```

$N := 5$

4.359227824

4.659227824

4.959227824

$N := 10$

4.514795679

4.664795680

4.814795679

$N := 20$

4.591198172

4.666198172

4.741198172

$N := 40$

4.629049495

4.666549494

4.704049495

$N := 80$

4.647887370

4.666637370

4.685387370

$N := 160$

4.657284343

4.666659344

4.676034343

$N := 320$

4.661977336

4.666664836

4.671352336

$N := 640$

4.664322459

4.666666208

4.669009959

$N := 1280$

4.665494677

4.666666551

4.667838427

$N := 2560$

4.666080701

4.666666638

4.667252576 **(3)**

>

>

>

Write commands that calculate the trapezoid and midpoint rules with **5, 10, 20, 40, 80, 160, 320, 640, 1280, 2560 subintervals. You can use a loop. What do you notice? Which are larger, smaller? Which are overestimates and which are underestimates of the integral? Why?**

```
> for k from 0 to 9 do N := 5·2^k; evalf(middlesum(f(x), x = 1 ..4, N)); evalf(0.5
    ·((leftsum(f(x), x = 1 ..4, N)) + (rightsum(f(x), x = 1 ..4, N)))); od;
```

$N := 5$

4.670363534

4.659227824

$N := 10$

4.667600664

$$4.664795680$$
$$N := 20$$
$$4.666900820$$
$$4.666198172$$
$$N := 40$$
$$4.666725247$$
$$4.666549494$$
$$N := 80$$
$$4.666681312$$
$$4.666637370$$
$$N := 160$$
$$4.666670329$$
$$4.666659344$$
$$N := 320$$
$$4.666667582$$
$$4.666664836$$
$$N := 640$$
$$4.666666896$$
$$4.666666208$$
$$N := 1280$$
$$4.666666725$$
$$4.666666551$$
$$N := 2560$$
$$4.666666680$$
$$4.666666638 \quad \textbf{(4)}$$

\>

\>

\>

Maple does not have a command to plot the trapezoid rule automatically, as it was the case for left-hand sum, right-hand sum and midpoint rule. But we can introduce a number of commands to see the graph. The following commands let Maple know of the lower limit, upper limit and the number of subintervals. The length of each subinterval is $\frac{b-a}{n}$. In the following example we choose

$n = 2.$

```
> a:=1;b:=4;n:=2;Dx:=(b-a)/n;
```

$$a := 1$$
$$b := 4$$
$$n := 2$$
$$Dx := \frac{3}{2}$$

We create a list of numbers in increasing order that represent the points between *a* and *b*, where we have split the interval [*a*,*b*]. In all we have $n + 1$ points.

```
> xpoints:=[seq( a+Dx*i, i=0..n)];
```

$$xpoints := \left[1, \frac{5}{2}, 4\right]$$

The next command finds the values of the function $f(x)$ at the points we are interested in.

```
> valuesoflist:=map(f, xpoints);
```

$$valuesoflist := \left[1, \frac{1}{2}\sqrt{10}, 2\right]$$

The next two commands pair together the *x* and *y* coordinates to form a list of points we would like to join with a broken line. We call this list datapoints.

```
> pair:=(x,y)->[x,y];
```

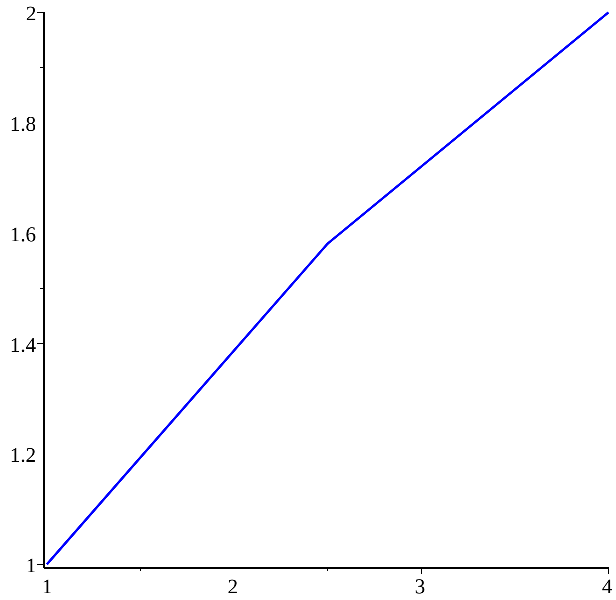
$$pair := (x, y) \rightarrow [x, y]$$

```
> datapoints:=zip(pair, xpoints,valuesoflist);
```

$$datapoints := \left[[1, 1], \left[\frac{5}{2}, \frac{1}{2}\sqrt{10}\right], [4, 2]\right]$$

The next command plots a broken line joining the points in the list datapoints.

```
> plot(datapoints, style=line, color=blue);
```

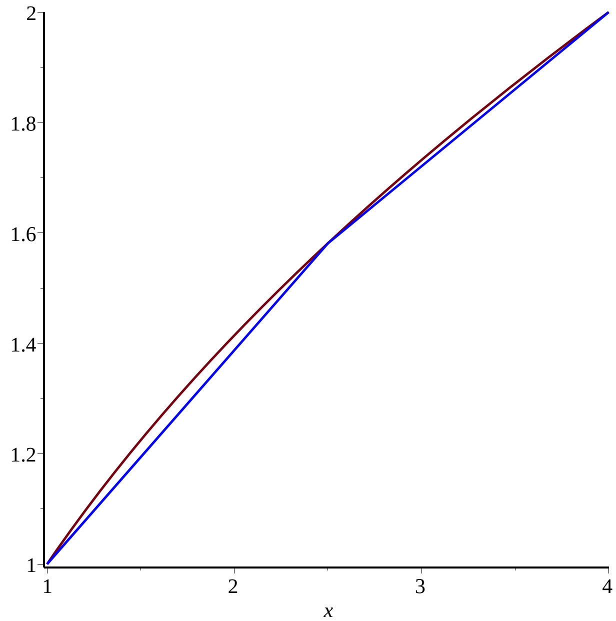


As usual it is nice to name this plot:

```
> trap2:=plot(datapoints, style=line, color=blue):
```

We now display the graph of $f(x)$ with the broken line we just saw.

```
> display(plot(f(x), x=1..4), trap2);
```



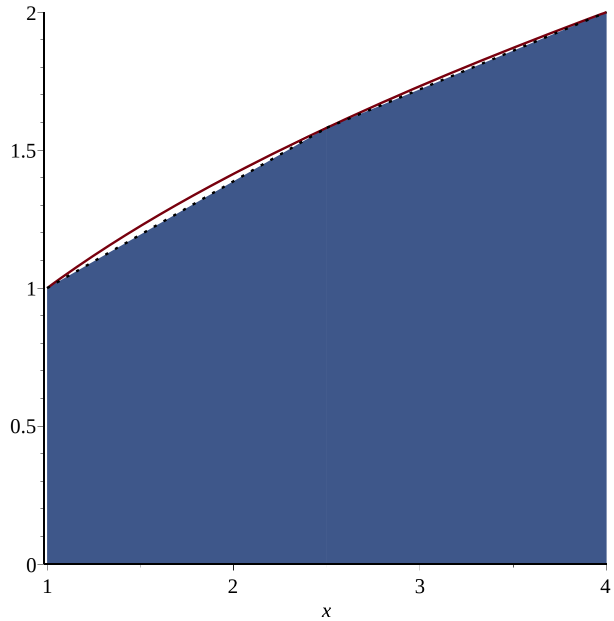
If we want to shade the area represented by the trapezoid rule, we use the following commands that shade the area under each part of the broken blue line and give a name to the corresponding graph: trapezoid[i]. We use a loop.

```
>  for i from 1 to n do trapezoid[i]:=inequal({y<f(xpoints[i])+(f
  (xpoints[i+1])-f(xpoints[i]))/Dx *(x-xpoints[i])}, x=xpoints[i]..
  xpoints[i+1], y=0..2, optionsexcluded=(color=white)): od;
```

$$trapezoid_1 := PLOT(\ldots)$$

$$trapezoid_2 := PLOT(\ldots)$$

```
> display(plot(f(x), x=1..4),trapezoid[1], trapezoid[2]);
```



We clearly see that the trapezoid rule in an underestimate.

Topic 4: The midpoint rule as a midpoint-tangent-trapezoid rule.

We need to graph the tangent line at the midpoints.

```
> midpoints:=[seq(a+Dx*(i-0.5), i=1..n)];
```

$$midpoints := [1.750000000, 3.250000000]$$

```
> derivatmidpoints:=map(D(f), midpoints);
```

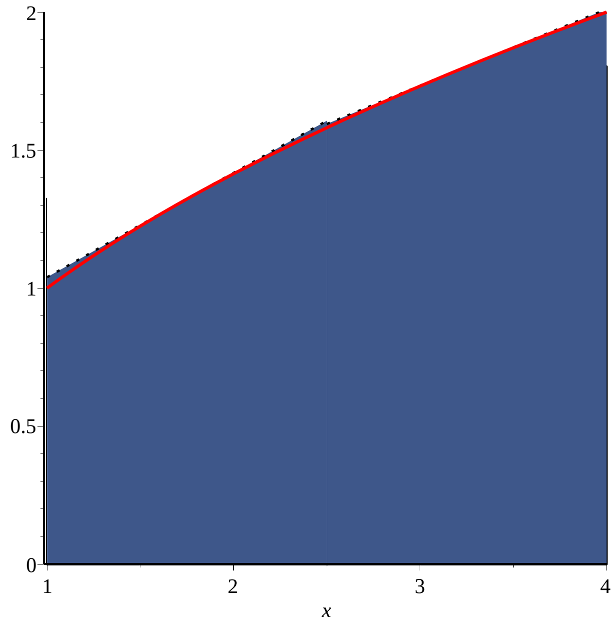
$$derivatmidpoints := [0.3779644729, 0.2773500980]$$

```
> for i from 1 to n do MID[i]:=inequal({y<f(midpoints[i])+
  derivatmidpoints[i]*(x-midpoints[i])}, x=xpoints[i]..xpoints
  [i+1], y=0..2, optionsexcluded=(color=white)): od;
```

$$MID_1 := PLOT(...)$$

$$MID_2 := PLOT(...)$$

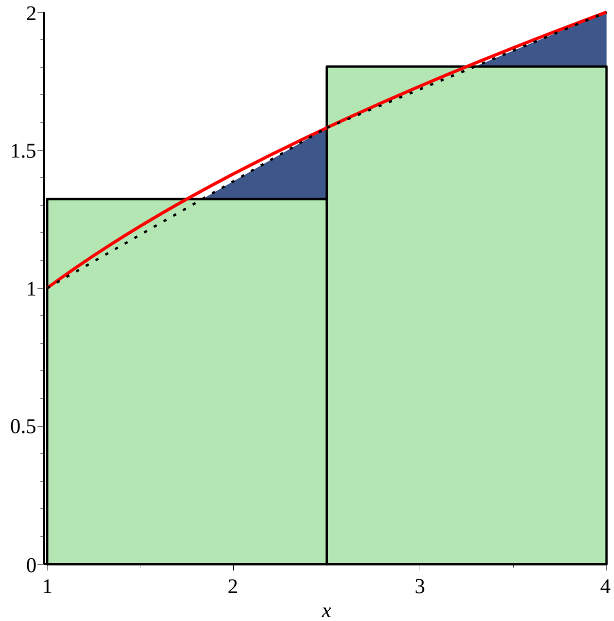
```
> display(plot(f(x), x=1..4), MID[1],MID[2], middlebox(f(x), x=1.
  .4, 2));
```



>

We see that the trapezoids with slanted side the tangent line at the midpoint cover exactly the same area as the midpoint sums. We also see that, because $f(x)$ is concave downwards, the tangent lines lie above the graph of $f(x)$, so the midpoint rule is an overestimate. **Write a command that shows the graph and the comparison of the trapezoid sum and the midpoint sum with** 2 **subintervals**.

```
> display(plot(f(x), x = 1..4), middlebox(f(x), x = 1..4, 2), trapezoid[1], trapezoid[2] );
```



>

>

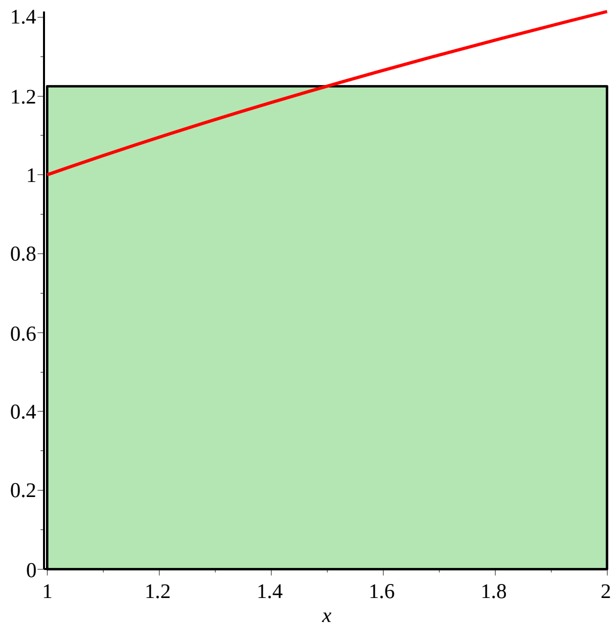
>

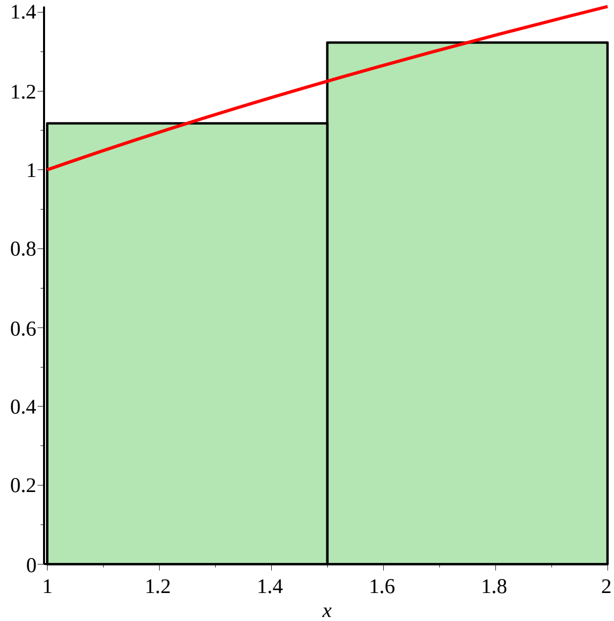
Work all the commands introduced today for the integral $\int_1^2 \frac{1}{x}\, dx$ with n=1, 2, 4, 8, 16, 32, 64 subintervals. Order in increasing order the left-hand sums, right-hand sums, midpoint suns and trapezoid sums. Explain you answer. Graph your sums for $n = 1$, $n = 2$ to explain you answers.

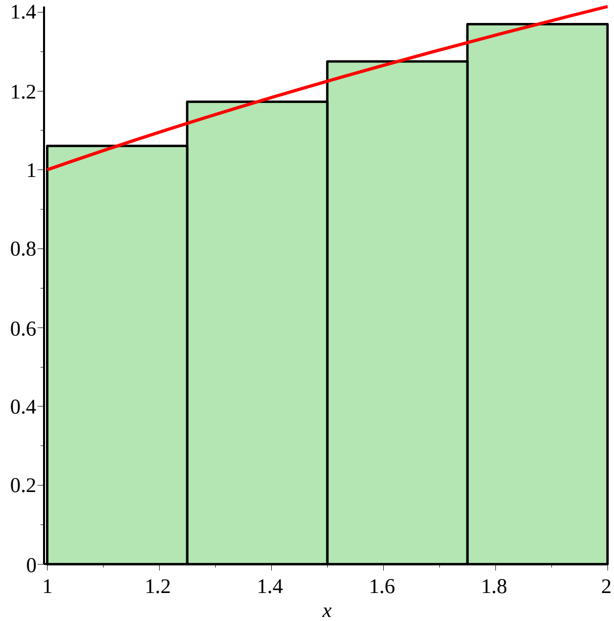
```
> g := x → sqrt(x);
```

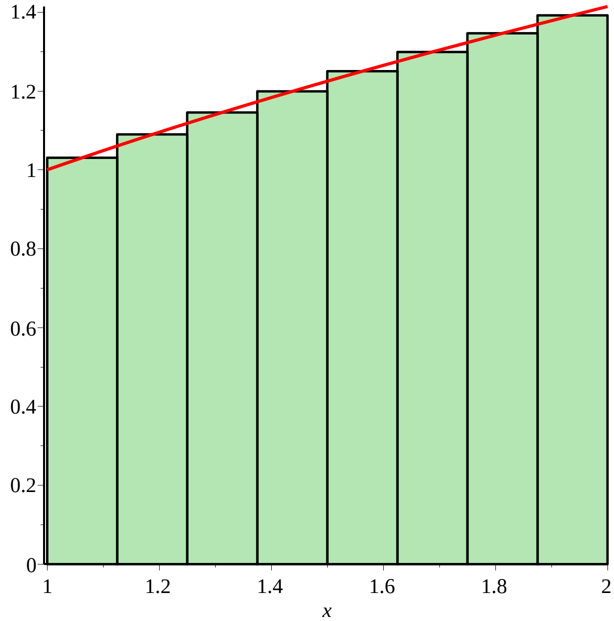
$$g := x \rightarrow \sqrt{x} \quad (5)$$

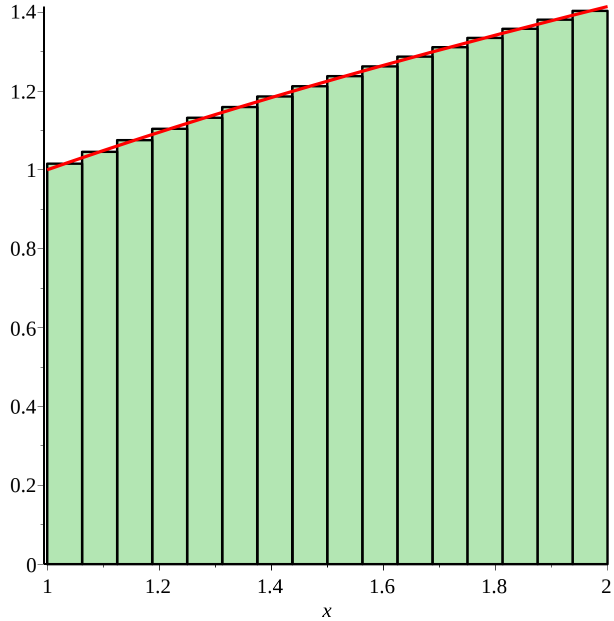
```
> middlebox(g(x), x = 1 ..2, 1); middlebox(g(x), x = 1 ..2, 2); middlebox(g(x), x = 1 ..2, 4);
    middlebox(g(x), x = 1 ..2, 8); middlebox(g(x), x = 1 ..2, 16); middlebox(g(x), x = 1 ..2,
    32); middlebox(g(x), x = 1 ..2, 64);
```

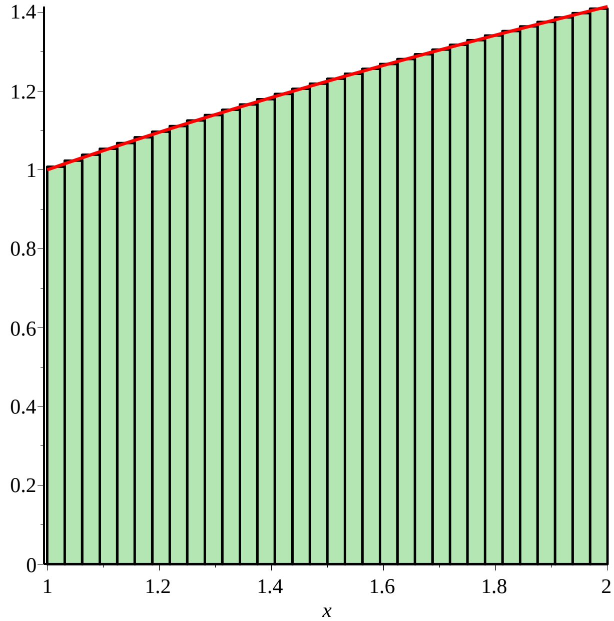

1.4
1.2
1
0.8
0.6
0.4
0.2
0
1
1.2
1.4
1.6
1.8
2
x

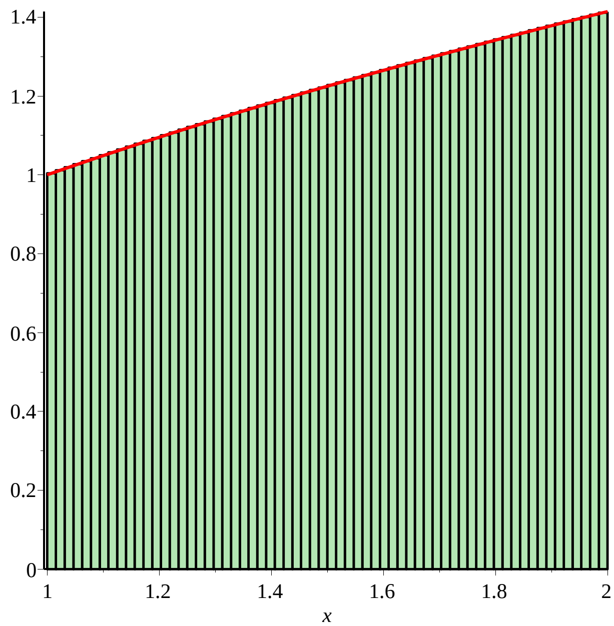

1.4
1.2
1
0.8
0.6
0.4
0.2
0
1
1.2
1.4
1.6
1.8
2
x


1.4
1.2
1
0.8
0.6
0.4
0.2
0
1
1.2
1.4
1.6
1.8
2
x


1.4
1.2
1
0.8
0.6
0.4
0.2
0
1
1.2
1.4
1.6
1.8
2
x


1.4
1.2
1
0.8
0.6
0.4
0.2
0
1
1.2
1.4
1.6
1.8
2
x


1.4
1.2
1
0.8
0.6
0.4
0.2
0
1
1.2
1.4
1.6
1.8
2
x




```
> msum1 := middlebox(g(x), x = 1 ..2, 1) : msum2 := middlebox(g(x), x = 1 ..2, 2) : msum4
      := middlebox(g(x), x = 1 ..2, 4) : msum8 := middlebox(g(x), x = 1 ..2, 8) : msum16
      := middlebox(g(x), x = 1 ..2, 16) : msum32 := middlebox(g(x), x = 1 ..2, 32) : msum64
      := middlebox(g(x), x = 1 ..2, 64) :
> evalf(middlesum(g(x), x = 1 ..2, 1)); evalf(middlesum(g(x), x = 1 ..2, 2));
     evalf(middlesum(g(x), x = 1 ..2, 4)); evalf(middlesum(g(x), x = 1 ..2, 8));
      evalf(middlesum(g(x), x = 1 ..2, 16)); evalf(middlesum(g(x), x = 1 ..2, 32));
       evalf(middlesum(g(x), x = 1 ..2, 64));
```

1.224744871

1.220454822

1.219331346

1.219046668

1.218975246

1.218957375

1.218952906 **(6)**

>