Note: In this problem set, expressions in green cells match corresponding expressions in the text answers.

General comment. I avoided the named methods such as Simpson’s, because they seemed merely quaint. I think that in terms of accuracy of presentation, it is not that hard to get Mathematica on the case.

```
Clear["Global`*"]
```

1 - 6 Rectangular and trapezoidal rules

1. Rectangular rule. Evaluate the integral in example 1 by the rectangular rule (1) with subintervals of length 0.1. Compare with example 1. (6S-exact: 0.746824).

```
Clear["Global`*"]
```

```
NIntegrate[e^(-x^2), {x, 0, 1}]
```

```
0.746824
```

Mathematica does not have a method for rectangular integration. The text answer implied the green above was an expected, accurate, or noteworthy result.

3. Trapezoidal rule. To get a feel for increase in accuracy, integrate x^2 from 0 to 1 by (2) with h=1, 0.5, 0.25, 0.1.

```
Clear["Global`*"]
```

```
N[Integrate[x^2, {x, 0, 1}], {3, 3}]
```

```
0.33
```

The above cell deliberately restricts the default accuracy and precision, just to show there is an effect from changing the numbers inside the curlies.

```
NIntegrate[x^2, {x, 0, 1}, Method -> "Trapezoidal"]
```

```
0.333333
```

```
NIntegrate[x^2, {x, 0, 1}, AccuracyGoal → 16,
 MaxRecursion → 500, WorkingPrecision → 10]
```

```
0.3333333333
```

4. Error estimation by halfing. Integrate f[x]=x^4 from 0 to 1 by (2) with h = 1, h = 0.5, h = 0.25, and estimate the error for h = 0.5 and h = 0.25 by (5).

5. Error estimation. Do the tasks in problem 4 for f[x]=Sin[$\frac{1}{2}\pi$ x].

```
Clear["Global`*"]
```

```
NIntegrate[Sin[1/2 π x], {x, 0, 1}]
```

0.63662

```
NIntegrate[Sin[1/2 π x], {x, 0, 1}, AccuracyGoal → 16,
 MaxRecursion → 500, WorkingPrecision → 10]
```

0.6366197724

Symbolab agrees with the answer, as far as it carries it. I can check this particular integral by hand.

```
Integrate[Sin[1/2 π x], x]
```

```
top = N[-(2 Cos[π x/2])/π, 16] /. x → 1
```

$0. \times 10^{-16}$

The format below is intended to require 16-digit accuracy with any precision.

```
bot = N[-(2 Cos[π x/2])/π, {∞, 16}] /. x → 0
```

-0.636619772367581

None of several tweaks I tried in this problem caused any change in the answer produced by Mathematica.

7 - 15 Simpson's rule

Evaluate the integrals A = Integrate[$\frac{1}{x}$,{x,1,2}] B=Integrate[x e^{-x^2} ,{x,0,0.4}] J=Integrate[$\frac{1}{1+x^2}$, {x,0,1}] by Simpson's rule as indicated, and compare with the exact value known from calculus.

7. A, 2m = 4

```
Integrate[1/x, x]
```

Log[x]

```
top = N[Log[2], {∞, 16}]
```

0.693147180559945

```
bot = N[Log[1], {∞, 16}]
```

$0. \times 10^{-16}$

Symbolab agrees with the answer, as far as it carries it. As a check,

```
e^0.693147180559945309374480365560700079l9`15.84082546104514
```

```
2.000000000000000
```

9. B, 2m = 4

```
rin = N[Integrate[x e^(-x^2) , {x, 0.0, 0.4}], {10, 16}]
```

```
0.0739281
```

```
rinn = NIntegrate[x e^(-x^2) , {x, 0.0, 0.4}, AccuracyGoal -> 16]
```

```
0.0739281
```

Symbolab agrees with the answer, as far as it carries it.

11. J, 2m = 4

```
NIntegrate[1/(1 + x^2) , {x, 0, 1}, AccuracyGoal -> 16]
```

```
0.785398
```

```
N[Integrate[1/(1 + x^2) , {x, 0, 1}], {10, 16}]
```

```
0.7853981634
```

Symbolab agrees with the answer, as far as it carries it.

13. Error estimate. Compute the integral J by Simpson’s rule with 2 m=8 and use the value and that in problem 11 to estimate the error by (10).

```
Clear["Global`*"]
```

Here I have made an effort to address estimated error, using material from the Mathematica documentation, *tutorial/NIntegrateIntegrationStrategies#285388386*, located at about 55% down the scroll. Some things I noticed. Both TrapStep modules are necessary. The improvement (decrease) in estimated local error occurs through adjustment of the “MaxRecursion” variable in the second module, which started at 7 with a fairly large estimated error on the integral. Increasing the value of “MaxRecursion” allows the variable tol_ to be decreased without triggering error messages, and the discovered error can go down. The calculation time goes up quite a bit with the increase in MaxRecursion, so it’s best to start low.

```
TrapStep[f_, {a_, b_}, n_?IntegerQ] :=
  Module[{h, absc, is},
   h = (b - a)/(n - 1);
   absc = Table[i, {i, a, b, h}];
   is = h * Total[MapAt[# / 2 &, f /@ absc, {{1}, {-1}}]];
   {is, ∞, n}
  ];

TrapStep[f_, {a_, b_}, {oldEstimate_, oldError_, oldn_}] :=
  Module[{n, h, absc, is},
   n = 2 oldn - 1;
   h = (b - a)/(n - 1);
   absc = Table[i, {i, a + h, b - h, 2 h}];
   is = h * Total[f /@ absc] + oldEstimate / 2;
   {is, Abs[is - oldEstimate], n}
  ];

Options[TrapezoidalIntegration] = {"MaxRecursion" → 20};
TrapezoidalIntegration[f_, {a_, b_}, tol_, opts___] :=
 Block[{maxrec, k = 0, temp},
  maxrec = "MaxRecursion" /. {opts} /. Options[TrapezoidalIntegration];
  NestWhile[((temp = TrapStep[f, {a, b}, #]) && k++ < maxrec) &,
    TrapStep[f, {a, b}, 5], #[[2]] > tol &][[1]];
  temp[[1]]
 ]

f[x_] := 1/(1 + x^2)

(* test function inluded with the tutorial: f[x_]:=
 1/π Cos[80 Sin[x]- x]*)

res = TrapezoidalIntegration[f, {0, 1}, 10^-12] // N
0.785398

NumberForm[%, {10, 10}]
0.7853981634
```

The number produced above agrees with the answer in problem 11.

```
exact = Integrate[f[x], {x, 0, 1}]
```

$$\frac{\pi}{4}$$

```
Abs[res - exact] / exact
```

1.92954×10^{-13}

The above checks Mathematica's accuracy in a different way than in problem 11, but no discrepancy is noted.

15. Given TOL. Find the smallest n in computing A (see problems 7 and 8) such that 5S-accuracy is guaranteed (a) by (4) in the use of (2), (b) by (9) in the use of (7).

If I do not clear variables, I can just continue using the error estimate modules from the previous problem.

$$g[x\_] := \frac{1}{x}$$

```
res = TrapezoidalIntegration[g, {1, 2}, 10^-5] // N
```

0.693148

```
NumberForm[%, {5, 5}]
```

0.69315

```
exact = Integrate[g[x], {x, 1, 2}]
```

Log[2]

```
Abs[res - exact] / exact
```

3.35904×10^{-10}

Yellow is the form with 5 significant digits. Altering the problem to suit the chosen algorithm, the equivalent question for this problem concerns the maximum possible requested level for the tol_ variable needed in order to guarantee 5S, and here it is found to be 10^{-5}. I see that even though “MaxRecursion” was not altered from the previous problem, the calculation speed increases dramatically with increase of the tol_ variable.

16 - 21 Nonelementary integrals

The following integrals cannot be evaluated by the usual methods of calculus. Evaluate them as indicated. Compare your value with that possibly given by your CAS. Si [x] is the sine integral. S[x] and C[x] are the Fresnel integrals. See appendix A3.1. They occur in optics.

$$\text{Si}[x] = \text{Integrate}\left[\frac{\text{Sin}[x^*]}{x^*}, \{x, 0, x\}\right]$$

$$\text{S}[x] = \text{Integrate}\left[\text{Sin}\left[x^{*2}\right], \{x, 0, x\}\right]$$

$$\text{C}[x] = \text{Integrate}\left[\text{Cos}\left[x^{*2}\right], \{x, 0, x\}\right]$$

17. Si[1] by (7), 2 m = 2, 2 m = 4

```
N[Integrate[Sin[x*]/x*, {x, 0, 1}], {10, 16}]
```

```
0.9460830704
```

19. Si[1] by (7), 2 m = 10

```
N[Integrate[Sin[x*]/x*, {x, 0, 1}], {7, 16}]
```

```
0.9460831
```

21. C[1.25] by numbered line (7), p. 832, 2 m = 10

```
NIntegrate[Cos[x*^2], {x, 0, 1.25}, PrecisionGoal → 14, AccuracyGoal → 16]
0.977438
```

```
N[Integrate[Cos[x*^2], {x, 0.0, 1.25}], {10, 16}]
0.977438
```

```
NumberForm[%, {7, 7}]
```

```
0.9774377
```

22 - 25 Gauss integration

Integrate by numbered line (11), p. 837, with n = 5:

23. x e^{-x} from 0 to 1

```
N[Integrate[x ⅇ^-x, {x, 0.0, 1.}], {10, 10}]
0.264241
```

```
NumberForm[%, {10, 10}]
```

```
0.2642411177
```

The number matches the text answer for 10S.

25. Exp[$-x^2$] from 0 to 1

```
N[Integrate[Exp[-x^2], {x, 0.0, 1.0}], {9, 9}]
0.746824
```

```
NumberForm[%, {9, 9}]
```

```
0.746824133
```

The number in the above cell matches the text answer for 9S.

27 - 30 Differentiation

27. Consider f[x] $= x^4$ for $x_0 = 0$, $x_1 = 0.2$, $x_2 = 0.4$, $x_3 = 0.6$, $x_4 = 0.8$. Calculate $f_2{}'$ from (14a), (14b), (14c), (15). Determine the errors. Compare and comment.

I'm going to skip the intended mechanics of the problem. I make a table of the expected values of the problem for reference.

```
f[x_] = x^4
x^4
```

```
Table[f'[x], {x, 0, 0.8, 0.2}]
{0., 0.032, 0.256, 0.864, 2.048}
```

There are various ways to get an approximate derivative by numerical means, and I look at three here.

1. ND

I see in reviewing some aspects of numerical differentiation that Mathematica has a built-in function for it, called ND. An extra package, not loaded by default, needs to be available to use what I might refer to as NumericalDerivative.

```
Needs["NumericalCalculus`"]
```

I use ND with a sample value, receiving back the expected result. The Scale parameter is used "to capture the region of variation", according to the documentation. With some functions such as sine, Scale can be set to zero, but with the current function it must be nonzero.

```
ND[f[x], x, 0.4, Scale → 0.0001, WorkingPrecision → 20]
```

```
0.256
```

```
NumberForm[%, {10, 10}]
0.2560000000
```

2. Definition

According to Wolfram *MathWorld*, the derivative definition is sometimes used for obtaining the numerical derivative, and that's what I do here.

```
fp[x_, h_] = (f[x + h] - f[x])/h
```

$$\frac{-x^4 + (h + x)^4}{h}$$

```
fp[0.4, 0.0001]
```

0.256096

The above looks useful. Additionally, I might want to look at a grid of the derivative values made using the problem's sample points juxtaposed against a list of function values using common values of h.

```
Grid[Table[Table[{4 x^3, fp[x, h]}, {x, 0, 0.8, 0.2}],
  {h, 0.0001, 0.001, 0.0001}], Frame → All]
```

{0., $1.\times10^{-12}$}	{0.032, 0.032024}	{0.256, 0.256096}	{0.864, 0.864216}	{2.048, 2.04838}
{0., $8.\times10^{-12}$}	{0.032, 0.032048}	{0.256, 0.256192}	{0.864, 0.864432}	{2.048, 2.04877}
{0., 2.7×10^{-11}}	{0.032, 0.0320721}	{0.256, 0.256288}	{0.864, 0.864648}	{2.048, 2.04915}
{0., 6.4×10^{-11}}	{0.032, 0.0320961}	{0.256, 0.256384}	{0.864, 0.864864}	{2.048, 2.04954}
{0., 1.25×10^{-10}}	{0.032, 0.0321202}	{0.256, 0.25648}	{0.864, 0.865081}	{2.048, 2.04992}
{0., 2.16×10^{-10}}	{0.032, 0.0321443}	{0.256, 0.256577}	{0.864, 0.865297}	{2.048, 2.05031}
{0., 3.43×10^{-10}}	{0.032, 0.0321684}	{0.256, 0.256673}	{0.864, 0.865513}	{2.048, 2.05069}
{0., 5.12×10^{-10}}	{0.032, 0.0321925}	{0.256, 0.256769}	{0.864, 0.86573}	{2.048, 2.05107}
{0., 7.29×10^{-10}}	{0.032, 0.0322166}	{0.256, 0.256865}	{0.864, 0.865946}	{2.048, 2.05146}
{0., $1.\times10^{-9}$}	{0.032, 0.0322408}	{0.256, 0.256962}	{0.864, 0.866162}	{2.048, 2.05184}

3. DifferenceDelta

Mathematica has a built-in function called DifferenceDelta that contains the functionality of the derivative definition. It goes like

```
(1/h) DifferenceDelta[f[x], {x, 1, h}]
```

$$\frac{-f[x] + f[h + x]}{h}$$

or for the problem function

```
(1/h) DifferenceDelta[x^4, {x, 1, h}]
```

$$\frac{h^4 + 4 h^3 x + 6 h^2 x^2 + 4 h x^3}{h}$$

and for the sample point already looked at

```
% /. {x → 0.4, h → 0.000001}
0.256001
```

It seems interesting that with ND, at Scale→0.0001, the answer already equals the exact, whereas with DifferenceDelta, at h→0.000001, there is still a little tail. Does this mean that DifferenceDelta is more exact, or that ND is more efficient?

29. The derivative $f'[x]$ can also be approximated in terms of first-order and higher order differences (see section 19.3):

$$f'[x_0] \approx \frac{1}{h}\left(\triangle f_0 - \frac{1}{2}\triangle^2 f_0 + \frac{1}{3}\triangle^2 f_0 - \frac{1}{4}\triangle f_0 + - \ldots\right).$$

Compute $f'[0.4]$ in problem 27 from this formula, using differences up to and including first order, second order, third order, fourth order.

I think this problem has been sufficiently covered in the discussion of the last problem.