Building and Using a Logarithmic Table

Helmut Knaust, Department of Mathematical Sciences, UTEP, El Paso TX 79968
hknaust@utep.edu
10/22/2011

How to compute Square Roots?

To build a logarithmic table, one must be able to compute square roots. How to do this was already kown to the Babylonians.

An example: To compute (rational) approximations to $\sqrt{10}$, one starts with a reasonable guess such as $x_0 = 3$.

Since x_0 =3 is too small $\left(x_0{}^2=9<10\right)$, $y_0 = \frac{10}{x_0}$ will be too large $\left(y_0{}^2 = \frac{100}{9}>10\right)$, so a new guess will be their average:

$x_1 = \frac{1}{2}\left(x_0 + \frac{10}{x_o}\right)$.

Computations using this algorithm are below. Note how fast the algorithm converges!

- **Square root of 10 :**

```
x[0] = 3;
x[n_] := 1 / 2 (x[n - 1] + 10 / x[n - 1])
TableForm[Table[{x[k], N[x[k], 20], N[x[k]^2, 20]}, {k, 0, 5}],
 TableHeadings → {None, {"xn", "xn", "xn2"}}]
```

x_n	x_n	$x_n{}^2$
3	3.0000000000000000000	9.0000000000000000000
$\frac{19}{6}$	3.1666666666666666667	10.027777777777777778
$\frac{721}{228}$	3.1622807017543859649	10.000019236688211757
$\frac{1\,039\,681}{328\,776}$	3.1622776601698420809	10.000000000009251237
$\frac{2\,161\,873\,163\,521}{683\,644\,320\,912}$	3.1622776601683793320	10.000000000000000000
$\frac{9\,347\,391\,150\,304\,592\,810\,234\,881}{2\,955\,904\,621\,546\,382\,351\,702\,304}$	3.1622776601683793320	10.000000000000000000

Iteration yields approximations for $\sqrt[4]{10}$, $\sqrt[8]{10}$, etc.

4th root of 10:

```
x[0] = 89 / 50;
x[n_] := 1 / 2 (x[n - 1] + 2 161 873 163 521 / 683 644 320 912 / x[n - 1])
TableForm[Table[{x[k], N[x[k], 20], N[x[k]^4, 20]}, {k, 0, 3}],
 TableHeadings → {None, {"xₙ", "xₙ", "xₙ⁴"}}]
```

x_n	x_n	$x_n{}^4$
$\frac{89}{50}$	1.7800000000000000000	10.038758560000000000
$\frac{2\,704\,957\,393\,686\,613}{1\,521\,108\,614\,029\,200}$	1.7782802416203312730	10.000018705316770566
$\frac{14\,633\,582\,160\,181\,403\,835\,379\,364\,781\,769}{8\,229\,067\,984\,237\,362\,013\,360\,062\,199\,200}$	1.7782794100391172384	10.000000000004373603
$\frac{428\,283\,453\,677\,512\,054\,374\,313\,233\,413\,617\,267\,850\,546\,843\,947\,786\,229\,696\,889\,361}{240\,841\,484\,898\,111\,612\,916\,211\,366\,908\,425\,588\,725\,736\,008\,781\,239\,548\,412\,769\,600}$	1.7782794100389228012	10.000000000000000000

Building a Logarithmic Table

Using just $\sqrt{10}$ yields a very primitive logarithmic table. The first row contains numbers between 1 and 10, the second row their logarithms with base 10.

```
Table[{10^x, x}, {x, 0, 1, 1 / 2}]
Table[{10^x, x}, {x, 0, 1, 1 / 2}] // N
```

$$\begin{pmatrix} 1 & 0 \\ \sqrt{10} & \frac{1}{2} \\ 10 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 1. & 0. \\ 3.16228 & 0.5 \\ 10. & 1. \end{pmatrix}$$

Using $\sqrt{10}$ and $\sqrt[4]{10}$ yields a slightly more detailed table:

```
Table[{10^x, x}, {x, 0, 1, 1 / 4}]
Table[{10^x, x}, {x, 0, 1, 1 / 4}] // N
```

$$\begin{pmatrix} 1 & 0 \\ \sqrt[4]{10} & \frac{1}{4} \\ \sqrt{10} & \frac{1}{2} \\ 10^{3/4} & \frac{3}{4} \\ 10 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 1. & 0. \\ 1.77828 & 0.25 \\ 3.16228 & 0.5 \\ 5.62341 & 0.75 \\ 10. & 1. \end{pmatrix}$$

We will use a much more detailed table, using $\sqrt[1024]{10}$. Here is a subset of our table:

```
tab = Transpose[Table[{10^x, x}, {x, 0, 1, 1 / 1024}] // N];
Transpose[
 Table[Table[{10^x, x}, {x, (k - 1) / 8, k / 8 - 1 / 1024, 2 / 1024}] // N, {k, 1, 8}]]
```

{1., 0.}	{1.33352, 0.125}	{1.77828, 0.25}	{2.37137, 0.375}	{3.16228, 0.5}	{4.21697, 0.625}	{5.62341, 0.75}	{7.49894, 0.875}
{1.00451, 0.00195313}	{1.33953, 0.126953}	{1.78629, 0.251953}	{2.38206, 0.376953}	{3.17653, 0.501953}	{4.23597, 0.626953}	{5.64876, 0.751953}	{7.53274, 0.876953}
{1.00904, 0.00390625}	{1.34557, 0.128906}	{1.79435, 0.253906}	{2.3928, 0.378906}	{3.19085, 0.503906}	{4.25507, 0.628906}	{5.67422, 0.753906}	{7.5667, 0.878906}
{1.01358, 0.00585938}	{1.35163, 0.130859}	{1.80243, 0.255859}	{2.40358, 0.380859}	{3.20523, 0.505859}	{4.27424, 0.630859}	{5.6998, 0.755859}	{7.6008, 0.880859}
{1.01815, 0.0078125}	{1.35773, 0.132813}	{1.81056, 0.257813}	{2.41442, 0.382813}	{3.21968, 0.507813}	{4.29351, 0.632813}	{5.72549, 0.757813}	{7.63506, 0.882813}
{1.02274, 0.00976563}	{1.36385, 0.134766}	{1.81872, 0.259766}	{2.4253, 0.384766}	{3.23419, 0.509766}	{4.31286, 0.634766}	{5.75129, 0.759766}	{7.66947, 0.884766}
{1.02735, 0.0117188}	{1.36999, 0.136719}	{1.82692, 0.261719}	{2.43623, 0.386719}	{3.24877, 0.511719}	{4.3323, 0.636719}	{5.77722, 0.761719}	{7.70404, 0.886719}
{1.03198, 0.0136719}	{1.37617, 0.138672}	{1.83515, 0.263672}	{2.44721, 0.388672}	{3.26341, 0.513672}	{4.35183, 0.638672}	{5.80326, 0.763672}	{7.73877, 0.888672}
{1.03663, 0.015625}	{1.38237, 0.140625}	{1.84342, 0.265625}	{2.45824, 0.390625}	{3.27812, 0.515625}	{4.37144, 0.640625}	{5.82942, 0.765625}	{7.77365, 0.890625}
{1.04131, 0.0175781}	{1.3886, 0.142578}	{1.85173, 0.267578}	{2.46932, 0.392578}	{3.2929, 0.517578}	{4.39115, 0.642578}	{5.85569, 0.767578}	{7.80869, 0.892578}
{1.046, 0.0195313}	{1.39486, 0.144531}	{1.86008, 0.269531}	{2.48045, 0.394531}	{3.30774, 0.519531}	{4.41094, 0.644531}	{5.88208, 0.769531}	{7.84389, 0.894531}
{1.05071, 0.0214844}	{1.40115, 0.146484}	{1.86846, 0.271484}	{2.49163, 0.396484}	{3.32265, 0.521484}	{4.43082, 0.646484}	{5.9086, 0.771484}	{7.87924, 0.896484}
{1.05545, 0.0234375}	{1.40746, 0.148438}	{1.87688, 0.273438}	{2.50287, 0.398438}	{3.33762, 0.523438}	{4.45079, 0.648438}	{5.93523, 0.773438}	{7.91476, 0.898438}
{1.06021, 0.0253906}	{1.41381, 0.150391}	{1.88534, 0.275391}	{2.51415, 0.400391}	{3.35267, 0.525391}	{4.47086, 0.650391}	{5.96198, 0.775391}	{7.95043, 0.900391}
{1.06499, 0.0273438}	{1.42018, 0.152344}	{1.89384, 0.277344}	{2.52548, 0.402344}	{3.36778, 0.527344}	{4.49101, 0.652344}	{5.98885, 0.777344}	{7.98627, 0.902344}
{1.06979, 0.0292969}	{1.42658, 0.154297}	{1.90238, 0.279297}	{2.53686, 0.404297}	{3.38296, 0.529297}	{4.51125, 0.654297}	{6.01585, 0.779297}	{8.02226, 0.904297}
{1.07461, 0.03125}	{1.43301, 0.15625}	{1.91095, 0.28125}	{2.5483, 0.40625}	{3.39821, 0.53125}	{4.53158, 0.65625}	{6.04296, 0.78125}	{8.05842, 0.90625}
{1.07945, 0.0332031}	{1.43947, 0.158203}	{1.91957, 0.283203}	{2.55978, 0.408203}	{3.41353, 0.533203}	{4.55201, 0.658203}	{6.0702, 0.783203}	{8.09474, 0.908203}
{1.08432, 0.0351563}	{1.44596, 0.160156}	{1.92822, 0.285156}	{2.57132, 0.410156}	{3.42891, 0.535156}	{4.57253, 0.660156}	{6.09756, 0.785156}	{8.13123, 0.910156}
{1.0892, 0.0371094}	{1.45248, 0.162109}	{1.93691, 0.287109}	{2.58291, 0.412109}	{3.44437, 0.537109}	{4.59314, 0.662109}	{6.12505, 0.787109}	{8.16788, 0.912109}
{1.09411, 0.0390625}	{1.45902, 0.164063}	{1.94564, 0.289063}	{2.59455, 0.414063}	{3.45989, 0.539063}	{4.61384, 0.664063}	{6.15265, 0.789063}	{8.2047, 0.914063}
{1.09905, 0.0410156}	{1.4656, 0.166016}	{1.95441, 0.291016}	{2.60625, 0.416016}	{3.47549, 0.541016}	{4.63464, 0.666016}	{6.18039, 0.791016}	{8.24168, 0.916016}
{1.104, 0.0429688}	{1.47221, 0.167969}	{1.96322, 0.292969}	{2.61799, 0.417969}	{3.49115, 0.542969}	{4.65553, 0.667969}	{6.20824, 0.792969}	{8.27883, 0.917969}
{1.10898, 0.0449219}	{1.47884, 0.169922}	{1.97207, 0.294922}	{2.62979, 0.419922}	{3.50689, 0.544922}	{4.67651, 0.669922}	{6.23623, 0.794922}	{8.31614, 0.919922}
{1.11397, 0.046875}	{1.48551, 0.171875}	{1.98096, 0.296875}	{2.64165, 0.421875}	{3.52269, 0.546875}	{4.69759, 0.671875}	{6.26434, 0.796875}	{8.35363, 0.921875}
{1.11899, 0.0488281}	{1.4922, 0.173828}	{1.98989, 0.298828}	{2.65356, 0.423828}	{3.53857, 0.548828}	{4.71876, 0.673828}	{6.29257, 0.798828}	{8.39128, 0.923828}
{1.12404, 0.0507813}	{1.49893, 0.175781}	{1.99885, 0.300781}	{2.66552, 0.425781}	{3.55452, 0.550781}	{4.74003, 0.675781}	{6.32093, 0.800781}	{8.4291, 0.925781}
{1.12911, 0.0527344}	{1.50569, 0.177734}	{2.00786, 0.302734}	{2.67753, 0.427734}	{3.57054, 0.552734}	{4.7614, 0.677734}	{6.34942, 0.802734}	{8.46709, 0.927734}
{1.13419, 0.0546875}	{1.51247, 0.179688}	{2.01691, 0.304688}	{2.6896, 0.429688}	{3.58664, 0.554688}	{4.78286, 0.679688}	{6.37804, 0.804688}	{8.50526, 0.929688}
{1.13931, 0.0566406}	{1.51929, 0.181641}	{2.02601, 0.306641}	{2.70172, 0.431641}	{3.6028, 0.556641}	{4.80442, 0.681641}	{6.40679, 0.806641}	{8.54359, 0.931641}
{1.14444, 0.0585938}	{1.52614, 0.183594}	{2.03514, 0.308594}	{2.7139, 0.433594}	{3.61904, 0.558594}	{4.82607, 0.683594}	{6.43567, 0.808594}	{8.5821, 0.933594}
{1.1496, 0.0605469}	{1.53302, 0.185547}	{2.04431, 0.310547}	{2.72613, 0.435547}	{3.63536, 0.560547}	{4.84782, 0.685547}	{6.46468, 0.810547}	{8.62079, 0.935547}
{1.15478, 0.0625}	{1.53993, 0.1875}	{2.05353, 0.3125}	{2.73842, 0.4375}	{3.65174, 0.5625}	{4.86968, 0.6875}	{6.49382, 0.8125}	{8.65964, 0.9375}
{1.15999, 0.0644531}	{1.54687, 0.189453}	{2.06278, 0.314453}	{2.75076, 0.439453}	{3.6682, 0.564453}	{4.89162, 0.689453}	{6.52309, 0.814453}	{8.69868, 0.939453}
{1.16522, 0.0664063}	{1.55384, 0.191406}	{2.07208, 0.316406}	{2.76316, 0.441406}	{3.68473, 0.566406}	{4.91367, 0.691406}	{6.55249, 0.816406}	{8.73788, 0.941406}
{1.17047, 0.0683594}	{1.56084, 0.193359}	{2.08142, 0.318359}	{2.77562, 0.443359}	{3.70134, 0.568359}	{4.93582, 0.693359}	{6.58202, 0.818359}	{8.77727, 0.943359}
{1.17574, 0.0703125}	{1.56788, 0.195313}	{2.0908, 0.320313}	{2.78813, 0.445313}	{3.71803, 0.570313}	{4.95807, 0.695313}	{6.61169, 0.820313}	{8.81683, 0.945313}
{1.18104, 0.0722656}	{1.57495, 0.197266}	{2.10022, 0.322266}	{2.80069, 0.447266}	{3.73479, 0.572266}	{4.98042, 0.697266}	{6.64149, 0.822266}	{8.85657, 0.947266}
{1.18637, 0.0742188}	{1.58204, 0.199219}	{2.10969, 0.324219}	{2.81332, 0.449219}	{3.75162, 0.574219}	{5.00286, 0.699219}	{6.67143, 0.824219}	{8.89649, 0.949219}
{1.19171, 0.0761719}	{1.58918, 0.201172}	{2.1192, 0.326172}	{2.826, 0.451172}	{3.76853, 0.576172}	{5.02541, 0.701172}	{6.7015, 0.826172}	{8.93659, 0.951172}
{1.19709, 0.078125}	{1.59634, 0.203125}	{2.12875, 0.328125}	{2.83874, 0.453125}	{3.78552, 0.578125}	{5.04807, 0.703125}	{6.7317, 0.828125}	{8.97687, 0.953125}
{1.20248, 0.0800781}	{1.60353, 0.205078}	{2.13835, 0.330078}	{2.85153, 0.455078}	{3.80258, 0.580078}	{5.07082, 0.705078}	{6.76205, 0.830078}	{9.01733, 0.955078}
{1.2079, 0.0820313}	{1.61076, 0.207031}	{2.14799, 0.332031}	{2.86438, 0.457031}	{3.81972, 0.582031}	{5.09368, 0.707031}	{6.79253, 0.832031}	{9.05798, 0.957031}
{1.21335, 0.0839844}	{1.61802, 0.208984}	{2.15767, 0.333984}	{2.87729, 0.458984}	{3.83693, 0.583984}	{5.11663, 0.708984}	{6.82314, 0.833984}	{9.09881, 0.958984}
{1.21881, 0.0859375}	{1.62531, 0.210938}	{2.16739, 0.335938}	{2.89026, 0.460938}	{3.85423, 0.585938}	{5.1397, 0.710938}	{6.8539, 0.835938}	{9.13982, 0.960938}
{1.22431, 0.0878906}	{1.63264, 0.212891}	{2.17716, 0.337891}	{2.90329, 0.462891}	{3.8716, 0.587891}	{5.16286, 0.712891}	{6.88479, 0.837891}	{9.18101, 0.962891}
{1.22983, 0.0898438}	{1.64, 0.214844}	{2.18697, 0.339844}	{2.91638, 0.464844}	{3.88905, 0.589844}	{5.18613, 0.714844}	{6.91582, 0.839844}	{9.2224, 0.964844}
{1.23537, 0.0917969}	{1.64739, 0.216797}	{2.19683, 0.341797}	{2.92952, 0.466797}	{3.90658, 0.591797}	{5.20951, 0.716797}	{6.94699, 0.841797}	{9.26396, 0.966797}
{1.24094, 0.09375}	{1.65482, 0.21875}	{2.20673, 0.34375}	{2.94273, 0.46875}	{3.92419, 0.59375}	{5.23299, 0.71875}	{6.97831, 0.84375}	{9.30572, 0.96875}
{1.24653, 0.0957031}	{1.66228, 0.220703}	{2.21668, 0.345703}	{2.95599, 0.470703}	{3.94188, 0.595703}	{5.25658, 0.720703}	{7.00976, 0.845703}	{9.34766, 0.970703}
{1.25215, 0.0976563}	{1.66977, 0.222656}	{2.22667, 0.347656}	{2.96931, 0.472656}	{3.95964, 0.597656}	{5.28027, 0.722656}	{7.04136, 0.847656}	{9.3898, 0.972656}
{1.25779, 0.0996094}	{1.67729, 0.224609}	{2.23671, 0.349609}	{2.9827, 0.474609}	{3.97749, 0.599609}	{5.30407, 0.724609}	{7.07309, 0.849609}	{9.43212, 0.974609}
{1.26346, 0.101563}	{1.68485, 0.226563}	{2.24679, 0.351563}	{2.99614, 0.476563}	{3.99542, 0.601563}	{5.32798, 0.726563}	{7.10497, 0.851563}	{9.47464, 0.976563}
{1.26916, 0.103516}	{1.69245, 0.228516}	{2.25692, 0.353516}	{3.00965, 0.478516}	{4.01343, 0.603516}	{5.35199, 0.728516}	{7.137, 0.853516}	{9.51734, 0.978516}
{1.27488, 0.105469}	{1.70008, 0.230469}	{2.26709, 0.355469}	{3.02321, 0.480469}	{4.03152, 0.605469}	{5.37612, 0.730469}	{7.16917, 0.855469}	{9.56024, 0.980469}
{1.28062, 0.107422}	{1.70774, 0.232422}	{2.27731, 0.357422}	{3.03684, 0.482422}	{4.04969, 0.607422}	{5.40035, 0.732422}	{7.20148, 0.857422}	{9.60333, 0.982422}
{1.2864, 0.109375}	{1.71544, 0.234375}	{2.28757, 0.359375}	{3.05053, 0.484375}	{4.06794, 0.609375}	{5.42469, 0.734375}	{7.23394, 0.859375}	{9.64662, 0.984375}
{1.2922, 0.111328}	{1.72317, 0.236328}	{2.29788, 0.361328}	{3.06428, 0.486328}	{4.08628, 0.611328}	{5.44914, 0.736328}	{7.26655, 0.861328}	{9.6901, 0.986328}
{1.29802, 0.113281}	{1.73094, 0.238281}	{2.30824, 0.363281}	{3.07809, 0.488281}	{4.1047, 0.613281}	{5.4737, 0.738281}	{7.2993, 0.863281}	{9.73377, 0.988281}
{1.30387, 0.115234}	{1.73874, 0.240234}	{2.31865, 0.365234}	{3.09196, 0.490234}	{4.1232, 0.615234}	{5.49838, 0.740234}	{7.3322, 0.865234}	{9.77765, 0.990234}
{1.30975, 0.117188}	{1.74658, 0.242188}	{2.3291, 0.367188}	{3.1059, 0.492188}	{4.14178, 0.617188}	{5.52316, 0.742188}	{7.36525, 0.867188}	{9.82172, 0.992188}
{1.31565, 0.119141}	{1.75445, 0.244141}	{2.33959, 0.369141}	{3.1199, 0.494141}	{4.16045, 0.619141}	{5.54805, 0.744141}	{7.39845, 0.869141}	{9.86599, 0.994141}
{1.32158, 0.121094}	{1.76236, 0.246094}	{2.35014, 0.371094}	{3.13396, 0.496094}	{4.17921, 0.621094}	{5.57306, 0.746094}	{7.4318, 0.871094}	{9.91046, 0.996094}
{1.32754, 0.123047}	{1.7703, 0.248047}	{2.36073, 0.373047}	{3.14809, 0.498047}	{4.19804, 0.623047}	{5.59818, 0.748047}	{7.46529, 0.873047}	{9.95513, 0.998047}

Here are some commands to look up the closest value in our table for a given number, and the two closest values (the “interpolation pair”), respectively.

```
lookupL[x_] := tab[[All, Position[Abs[tab[[1]] - x], Min[Abs[tab[[1]] - x]]][[1, 1]]]]
lookupL[7.008]
```

{7.00976, 0.845703}

```
IntPairL[x_] :=
 {tab[[All, Position[tab[[1]], Max[Select[tab[[1]], # ≤ x &]]][[1, 1]]]],
  tab[[All, Position[tab[[1]], Min[Select[tab[[1]], # > x &]]][[1, 1]]]]}
IntPairL[7.018]
```

$$\begin{pmatrix} 7.00976 & 0.845703 \\ 7.02554 & 0.84668 \end{pmatrix}$$

```
lookupE[x_] := tab[[All, Position[Abs[tab[[2]] - x], Min[Abs[tab[[2]] - x]]][[1, 1]]]]
lookupE[.7]
```

{5.01413, 0.700195}

```
IntPairE[x_] :=
 {tab[[All, Position[tab[[2]], Max[Select[tab[[2]], # ≤ x &]]][[1, 1]]]],
  tab[[All, Position[tab[[2]], Min[Select[tab[[2]], # > x &]]][[1, 1]]]]}
IntPairE[.506]
```

$$\begin{pmatrix} 3.20523 & 0.505859 \\ 3.21245 & 0.506836 \end{pmatrix}$$

Characteristic and Mantissa

A logarithmic table only contains logarithms for numbers between 1 and 10.
Logarithms for other positive numbers can be found by using the fact that $\log(10^n x)=n + \log(x)$.
Here is an example. Our table yields 0.845703 as the log of 7.00976.
Consequently 1.845703 is the log of 70.0976, 0.845703-1 is the log of 0.700976.
Instead of log(0.700976)= 0.845703-1, one writes $\log(0.700976)=\bar{1}.845703$.
With this convention, the number in front of the decimal point is called “characteristic”, the part after the decimal point is called “mantissa”.
log(0.700976) has characteristic -1 and mantissa 845703.

Multiplication and Division

- Example: 2 · 3 = 6, using log (6) = log (2) + log (3)

  We first look up values for log 2 and log 3, then add them:

```
lookupL[2]
lookupL[3]
```

```
{1.99885, 0.300781}
```

```
{3.00289, 0.477539}
```

```
sum = lookupL[2][[2]] + lookupL[3][[2]]
```

```
0.77832
```

  Now we look up which value in the table has this logarithm:

```
lookupE[sum]
```

```
{6.00234, 0.77832}
```

  So 2 · 3 is approximately equal to:

```
lookupE[sum][[1]]
```

```
6.00234
```

- Example: 356.4 ÷ 1.87 = 190.588, using log (190.588) = log (356.4) - log (1.87)

  We look up values for log 3.564 and log 1.87:

```
lookupL[3.564]
lookupL[1.87]
```

```
{3.56252, 0.551758}
```

```
{1.86846, 0.271484}
```

  Since log 3.564 ≈ 0.551758, log 356.4 ≈ 2.551758.
  Subtraction yields:

```
diff = 2 + lookupL[3.564][[2]] - lookupL[1.87][[2]]
```

2.28027

Then we look up which value in the table has this mantissa:

```
lookupE[diff - 2]
```

{1.90666, 0.280273}

Multiplying by 10^2 yields the approximative answer

```
10^2 lookupE[diff - 2][[1]]
```

190.666

How to compute $3^{1.41}$?

- $\log\left(3^{1.41}\right) = 1.41 \log 3.$

```
lookupL[3]
```

{3.00289, 0.477539}

```
p1 = 1.41 lookupL[3][[2]]
```

0.67333

Final answer :

```
lookupE[p1]
lookupE[p1][[1]]
```

{4.70816, 0.672852}

4.70816

Here is Mathematica's answer:

```
3^1.41
```

4.70697

How to compute $7^{1.6}$?

- $\log\left(7^{1.6}\right) = 1.6 \log 7.$

```
lookupL[7]
```

{6.99402, 0.844727}

```
1.6 lookupL[7][[2]]
```

1.35156

The mantissa of this logarithm is

```
mant = 1.6 lookupL[7][[2]] - 1
```

0.351563

```
lookupE[mant]
```

{2.24679, 0.351563}

Final answer :

```
10 lookupE[mant][[1]]
```

22.4679

Mathematica gets:

```
7^1.6
```

22.4987

Linear Interpolation

We can get better results by using linear interpolation.
Let us redo the last example: $7^{1.6}$.
Below are the two entries closest to 7:

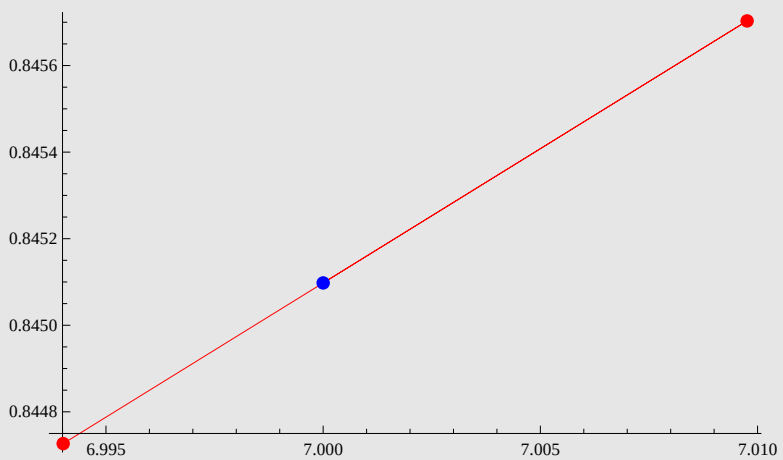
```
{pt1, pt2} = IntPairL[7]
```

$$\begin{pmatrix} 6.99402 & 0.844727 \\ 7.00976 & 0.845703 \end{pmatrix}$$

Assuming that the log function is linear between these values, we can get a better approximation for log 7:

```
ptc = {7, Fit[{pt1, pt2}, {1, x}, x] /. x → 7}
ListPlot[{pt1, pt2, ptc}, PlotStyle → Red, Joined → True,
 Epilog -> {{AbsolutePointSize[7], Red, Point[pt1], Point[pt2]},
   {AbsolutePointSize[7], Blue, Point[ptc]}}]
```

{7, 0.845098}




```
1.6 ptc[[2]]
```

1.35216

Subtract 1 to obtain the mantissa:

```
1.6 ptc[[2]] - 1
```

0.352156

Next we look up the interpolation pair for this mantissa:

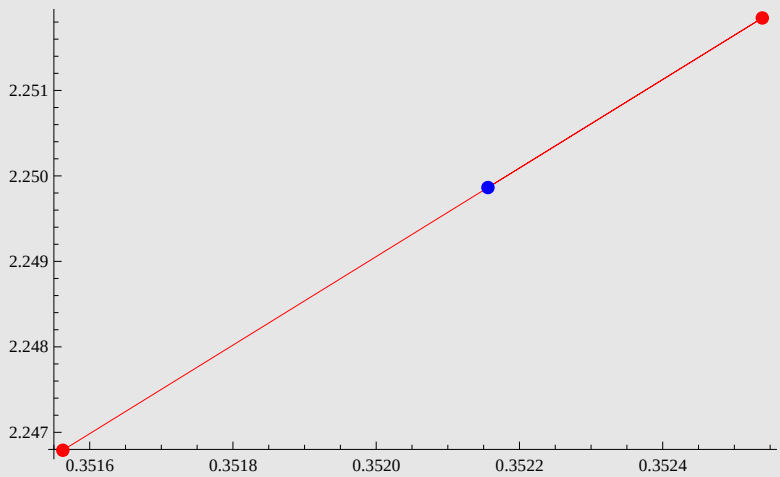
```
{pt11, pt12} = Map[Reverse[#] &, IntPairE[0.352156]]
```

$$\begin{pmatrix} 0.351563 & 2.24679 \\ 0.352539 & 2.25185 \end{pmatrix}$$

Linear interpolation yields:

```
ptc2 = {0.352156, Fit[{pt11, pt12}, {1, x}, x] /. x → 0.352156}
ListPlot[{pt11, pt12, ptc2}, Joined → True, PlotStyle → Red,
 Epilog -> {{AbsolutePointSize[7], Red, Point[pt11], Point[pt12]},
   {AbsolutePointSize[7], Blue, Point[ptc2]}}]
```

{0.352156, 2.24986}



Final answer :

```
10 ptc2[[2]]
```

22.4986

Mathematica' s answer :

$7^{1.6}$

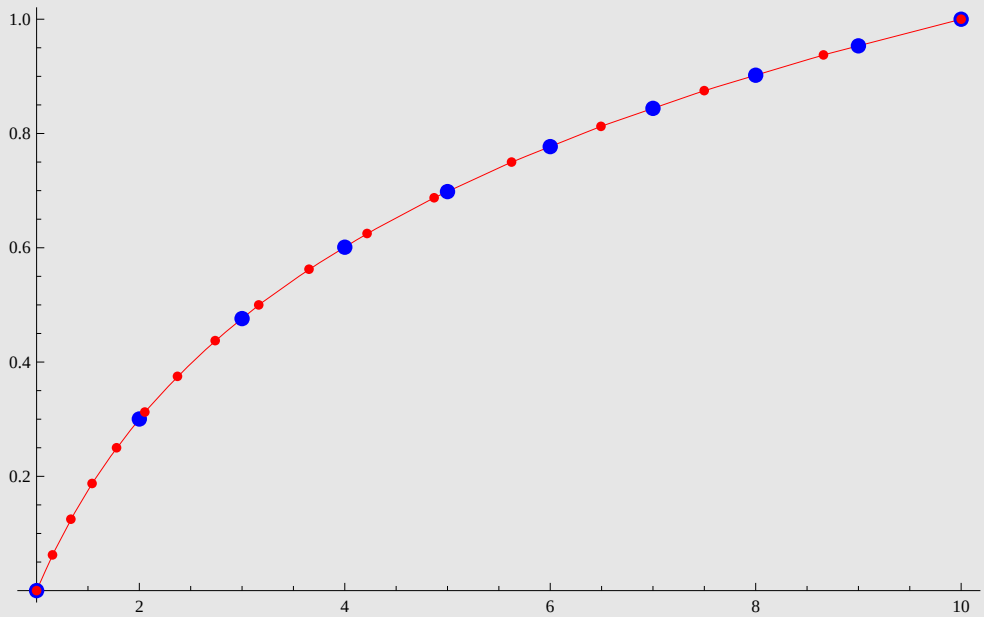
22.4987

Fixing the Log Table

Linear interpolation is also used to “refine” the logarithmic table itself.
The objective is to produce a table where the entries on the left are in arithmetic progression (constant difference between consecutive terms).

The example below shows how to produce a table of 10 entries in arithmetic progression from a raw logarithmic table with 17 entries (multiples of $\sqrt[16]{10}$).
The original values are depicted in red, the interpolated values in blue.

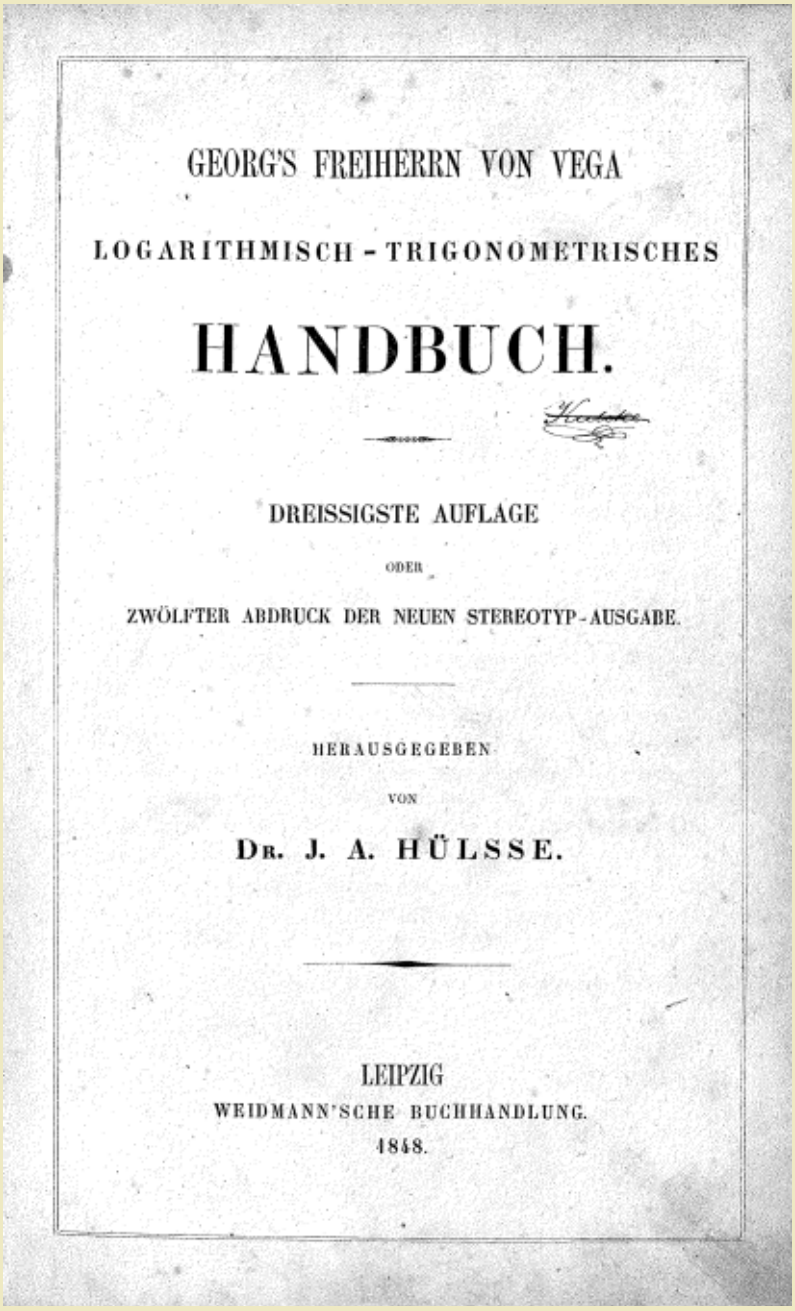
```
tabb = Table[{10^k, k}, {k, 0, 1, 1 / 16}];
f = Interpolation[tabb, InterpolationOrder → 1];
Plot[f[x], {x, 1, 10}, PlotStyle → Red,
 Epilog → {{Blue, AbsolutePointSize[8], Point[Table[{k, f[k]}, {k, 1, 10, 1}]]},
   {Red, AbsolutePointSize[5], Point[tabb]}}]
```




```
Table[{x, Chop[f[x]]}, {x, 1., 10, 1}]
```

$$\begin{pmatrix} 1. & 0 \\ 2. & 0.300346 \\ 3. & 0.476071 \\ 4. & 0.601009 \\ 5. & 0.698307 \\ 6. & 0.777041 \\ 7. & 0.843975 \\ 8. & 0.90198 \\ 9. & 0.953371 \\ 10. & 1. \end{pmatrix}$$

A German Logarithmic Table



GEORG'S FREIHERRN VON VEGA

LOGARITHMISCH - TRIGONOMETRISCHES

HANDBUCH.

DREISSIGSTE AUFLAGE

ODER

ZWÖLFTER ABDRUCK DER NEUEN STEREOTYP-AUSGABE.

HERAUSGEGEBEN

VON

DR. J. A. HÜLSSE.

LEIPZIG

WEIDMANN'SCHE BUCHHANDLUNG.

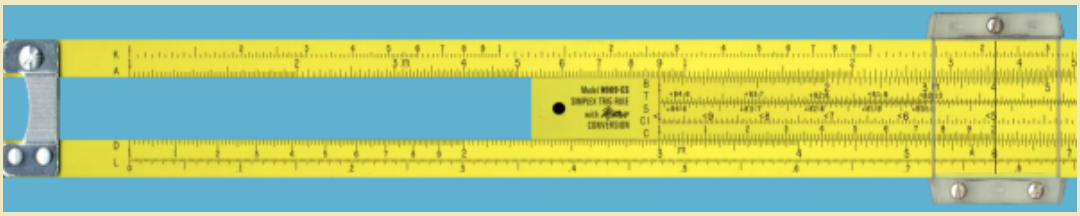
1848.

Note the built-in interpolation table on the right side of the page below. The differences between consecutive entries in the table are mostly (0.0000)163 at the top, and then become 162 and 161 further down.

N. 26500. L. 423.

N.	0	1	2	3	4	5	6	7	8	9	P.	P.
2650	423 2459	2623	2786	2950	3114	3278	3442	3606	3770	3933		163
2651	4097	4261	4425	4589	4753	4916	5080	5244	5408	5571	1	16
2652	5735	5899	6063	6226	6390	6554	6718	6881	7045	7209	2	33
2653	7372	7536	7700	7864	8027	8191	8355	8518	8682	8846	3	49
2654	9009	9173	9336	9500	9664	9827	9991	·0154	·0318	·0482	4	65
2655	424 0645	0809	0972	1136	1300	1463	1627	1790	1954	2117	5	82
2656	2281	2444	2608	2771	2935	3098	3262	3425	3589	3752	6	98
2657	3916	4079	4242	4406	4569	4733	4896	5060	5223	5386	7	114
2658	5550	5713	5877	6040	6203	6367	6530	6693	6857	7020	8	130
2659	7183	7347	7510	7673	7837	8000	8163	8327	8490	8653	9	147
2660	8816	8980	9143	9306	9469	9633	9796	9959	·0122	·0286		
2661	425 0449	0612	0775	0938	1102	1265	1428	1591	1754	1917		
2662	2081	2244	2407	2570	2733	2896	3059	3222	3385	3549		
2663	3712	3875	4038	4201	4364	4527	4690	4853	5016	5179		
2664	5342	5505	5668	5831	5994	6157	6320	6483	6646	6809		
2665	6972	7135	7298	7461	7624	7787	7950	8113	8276	8439		
2666	8601	8764	8927	9090	9253	9416	9579	9742	9904	·0067		
2667	426 0230	0393	0556	0719	0881	1044	1207	1370	1533	1695		
2668	1858	2021	2184	2347	2509	2672	2835	2998	3160	3323		
2669	3486	3648	3811	3974	4137	4299	4462	4625	4787	4950		
2670	5113	5275	5438	5601	5763	5926	6088	6251	6414	6576		162
2671	6739	6901	7064	7227	7389	7552	7714	7877	8039	8202	1	16
2672	8365	8527	8690	8852	9015	9177	9340	9502	9665	9827	2	32
2673	9990	·0152	·0315	·0477	·0639	·0802	·0964	·1127	·1289	·1452	3	49
2674	427 1614	1776	1939	2101	2264	2426	2588	2751	2913	3076	4	65
2675	3238	3400	3563	3725	3887	4050	4212	4374	4536	4699	5	81
2676	4861	5023	5186	5348	5510	5672	5835	5997	6159	6321	6	97
2677	6484	6646	6808	6970	7133	7295	7457	7619	7781	7944	7	113
2678	8106	8268	8430	8592	8754	8917	9079	9241	9403	9565	8	130
2679	9727	9889	·0051	·0213	·0376	·0538	·0700	·0862	·1024	·1186	9	146
2680	428 1348	1510	1672	1834	1996	2158	2320	2482	2644	2806		
2681	2968	3130	3292	3454	3616	3778	3940	4102	4264	4426		
2682	4588	4750	4912	5073	5235	5397	5559	5721	5883	6045		
2683	6207	6369	6530	6692	6854	7016	7178	7340	7501	7663		
2684	7825	7987	8149	8311	8472	8634	8796	8958	9119	9281		
2685	9443	9605	9766	9928	·0090	·0252	·0413	·0575	·0737	·0898		
2686	429 1060	1222	1383	1545	1707	1868	2030	2192	2353	2515		
2687	2677	2838	3000	3162	3323	3485	3646	3808	3969	4131		
2688	4293	4454	4616	4777	4939	5100	5262	5423	5585	5747		
2689	5908	6070	6231	6393	6554	6715	6877	7038	7200	7361		
2690	7523	7684	7846	8007	8169	8330	8491	8653	8814	8976		161
2691	9137	9298	9460	9621	9782	9944	·0105	·0267	·0428	·0589	1	16
2692	430 0751	0912	1073	1235	1396	1557	1718	1880	2041	2202	2	32
2693	2364	2525	2686	2847	3009	3170	3331	3492	3653	3815	3	48
2694	3976	4137	4298	4460	4621	4782	4943	5104	5265	5427	4	64
2695	5588	5749	5910	6071	6232	6393	6554	6716	6877	7038	5	81
2696	7199	7360	7521	7682	7843	8004	8165	8326	8487	8648	6	97
2697	8809	8970	9132	9293	9454	9615	9776	9937	·0098	·0258	7	113
2698	431 0419	0580	0741	0902	1063	1224	1385	1546	1707	1868	8	129
2699	2029	2190	2351	2512	2672	2833	2994	3155	3316	3477	9	145
N.	0	1	2	3	4	5	6	7	8	9	Diff.	

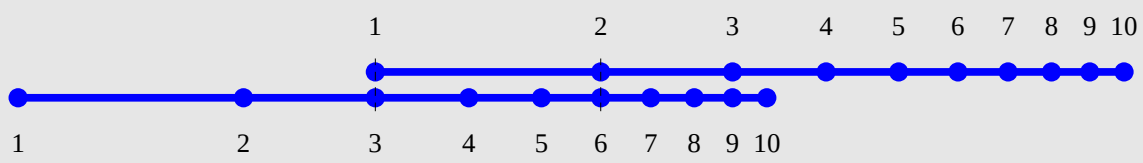
Slide Rules



Slide rules were used to multiply and divide quickly. The scale on the two rulers is chosen such that the distance between 1 and x is log x.

Below is the example how to multiply 3 by 2: We need to add the lengths log 3 and log 2 to obtain the length log 6.

```
Show[Graphics[{{AbsoluteThickness[4], Blue, Line[{{0, 0}, {1, 0}}]},
   {AbsolutePointSize[10], Blue, Point[Table[{Log[10, k], 0}, {k, 1, 10}]]},
   {Table[Text[k, {Log[10, k], 0}, {0, 3}], {k, 1, 10}]}, {AbsoluteThickness[4],
    Blue, Line[{{Log[10, 3], 1}, {1 + Log[10, 3], 1}}]}, {AbsolutePointSize[10],
    Blue, Point[Table[{Log[10, 3] + Log[10, k], 1}, {k, 1, 10}]]},
   {Table[Text[k, {Log[10, 3] + Log[10, k], 1}, {0, -3}], {k, 1, 10}]},
   {Dashed, Line[{{Log[10, 3], -0.5}, {Log[10, 3], 1.5}}]},
   {Dashed, Line[{{Log[10, 6], -0.5}, {Log[10, 6], 1.5}}]}}, BaseStyle → 14],
 ImageSize → {600, 100}, AspectRatio → 1 / 12]
```



A working model of a slide rule (Pickett N909-ES) can be found at http://www.antiquark.com/sliderule/sim/n909es/virtual-n909-es.html