To help you better understand why the definition of Big-O is concerned only with the behavior of functions for large values of *n*, choose two functions with different growth rates in which the faster growing function is lower at small values of *n*, but eventually becomes larger. Write a short program that periodically compares the values of the two functions and illustrates the point at which the faster growing function overtakes the slower growing one. As an example, consider the following two functions:

* *f*(*n*) = 500*n*2 + 15*n* + 1000
* *g*(*n*) = 2*n*3

Shown below is a table of the values of both functions for small values of *n*.

n f(n) g(n)

10 51150 2000

20 201300 16000

30 451450 54000

40 801600 128000

50 1251750 250000

60 1801900 432000

70 2452050 686000

80 3202200 1024000

90 4052350 1458000

100 5002500 2000000

110 6052650 2662000

120 7202800 3456000

130 8452950 4394000

140 9803100 5488000

150 11253250 6750000

160 12803400 8192000

170 14453550 9826000

180 16203700 11664000

190 18053850 13718000

200 20004000 16000000

210 22054150 18522000

220 24204300 21296000

230 26454450 24334000

240 28804600 27648000

250 31254750 31250000

260 33804900 35152000

Once *n* reaches 260 *g* overtakes *f*.