

```

1 ; vim: ts=2 sw=2 et :
2 ;
3 ;
4 ;
5 ;
6 ;
7 ;
8 ;
9 ;
10 ;
11 ;
12 ;
13 ;
14 ;
15 ;
16 ;
17 ;
18 (defvar *options* ' (
19   about "brknbad: explore the world better, explore the world for good.
20   (c) 2022, Tim Menzies
21
22   OPTIONS: "
23   cautious ("c" "abort on any error" t)
24   dump ("d" "stack dumps on error" nil)
25   enough ("e" "enough items for a sample" 512)
26   far ("f" "far away" .9)
27   file ("f" "read data from file" "../data/aut093.csv")
28   help ("h" "show help" nil)
29   license ("l" "show license" nil)
30   p ("p" "euclidean coefficient" 2)
31   seed ("s" "random number seed" 10019)
32   todo ("t" "start up action" "nothing"))
33
34 ; Copyright (c) 2021 Tim Menzies
35 ;
36 ; This is free and unencumbered software released into the public domain.
37 ;
38 ; Anyone is free to copy, modify, publish, use, compile, sell, or
39 ; distribute this software, either in source code form or as a compiled
40 ; binary, for any purpose, commercial or non-commercial, and by any
41 ; means.
42 ;
43 ; In jurisdictions that recognize copyright laws, the author or authors
44 ; of this software dedicate any and all copyright interest in the
45 ; software to the public domain. We make this dedication for the benefit
46 ; of the public at large and to the detriment of our heirs and
47 ; successors. We intend this dedication to be an overt act of
48 ; relinquishment in perpetuity of all present and future rights to this
49 ; software under copyright law.
50 ;
51 ; THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
52 ; EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
53 ; MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.
54 ; IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY CLAIM, DAMAGES OR
55 ; OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE,
56 ; ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR
57 ; OTHER DEALINGS IN THE SOFTWARE.
58 ;
59 ; For more information, please refer to <http://unlicense.org/>
60 ;
61 ;
62 ;
63 ;
64 ;
65 ;
66 ;
67 ;
68 ;
69 ;
70 ;

```



```

111 ;
112 ;
113 ;
114 ;
115 ;
116 ;
117 ;
118 ;
119 ;
120 ;
121 ;
122 ;
123 ;
124 ;
125 ;
126 ;
127 ;
128 ;
129 ;
130 ;
131 ;
132 ;
133 ;
134 ;
135 ;
136 ;
137 ;
138 ;
139 ;
140 ;
141 ;
142 ;
143 ;
144 ;
145 ;
146 ;
147 ;
148 ;
149 ;
150 ;
151 ;
152 ;
153 ;
154 ;
155 ;
156 ;
157 ;
158 ;
159 ;
160 ;
161 ;
162 ;
163 ;
164 ;
165 ;
166 ;
167 ;
168 ;
169 ;
170 ;
171 ;
172 ;
173 ;
174 ;
175 ;
176 ;
177 ;
178 ;
179 ;
180 ;
181 ;
182 ;
183 ;
184 ;
185 ;
186 ;
187 ;
188 ;
189 ;
190 ;
191 ;
192 ;
193 ;
194 ;
195 ;
196 ;
197 ;
198 ;
199 ;
200 ;
201 ;
202 ;
203 ;
204 ;
205 ;
206 ;
207 ;
208 ;
209 ;
210 ;
211 ;
212 ;
213 ;
214 ;
215 ;
216 ;
217 ;
218 ;
219 ;
220 ;
221 ;
222 ;
223 ;
224 ;
225 ;
226 ;
227 ;
228 ;
229 ;
230 ;
231 ;
232 ;
233 ;
234 ;
235 ;
236 ;
237 ;
238 ;
239 ;
240 ;
241 ;
242 ;
243 ;
244 ;
245 ;
246 ;
247 ;
248 ;
249 ;
250 ;
251 ;
252 ;
253 ;
254 ;
255 ;
256 ;
257 ;
258 ;
259 ;
260 ;
261 ;
262 ;
263 ;
264 ;
265 ;
266 ;
267 ;
268 ;
269 ;
270 ;
271 ;
272 ;
273 ;
274 ;
275 ;
276 ;
277 ;
278 ;
279 ;
280 ;
281 ;
282 ;
283 ;
284 ;
285 ;
286 ;
287 ;
288 ;
289 ;
290 ;
291 ;
292 ;
293 ;
294 ;
295 ;
296 ;
297 ;
298 ;
299 ;
300 ;
301 ;
302 ;
303 ;
304 ;
305 ;
306 ;
307 ;
308 ;
309 ;
310 ;
311 ;
312 ;
313 ;
314 ;
315 ;
316 ;
317 ;
318 ;
319 ;
320 ;
321 ;
322 ;
323 ;
324 ;
325 ;
326 ;
327 ;
328 ;
329 ;
330 ;
331 ;
332 ;
333 ;
334 ;
335 ;
336 ;
337 ;
338 ;
339 ;
340 ;
341 ;
342 ;
343 ;
344 ;
345 ;
346 ;
347 ;
348 ;
349 ;
350 ;
351 ;
352 ;
353 ;
354 ;
355 ;
356 ;
357 ;
358 ;
359 ;
360 ;
361 ;
362 ;
363 ;
364 ;
365 ;
366 ;
367 ;
368 ;
369 ;
370 ;
371 ;
372 ;
373 ;
374 ;
375 ;
376 ;
377 ;
378 ;
379 ;
380 ;
381 ;
382 ;
383 ;
384 ;
385 ;
386 ;
387 ;
388 ;
389 ;
390 ;
391 ;
392 ;
393 ;
394 ;
395 ;
396 ;
397 ;
398 ;
399 ;
400 ;
401 ;
402 ;
403 ;
404 ;
405 ;
406 ;
407 ;
408 ;
409 ;
410 ;
411 ;
412 ;
413 ;
414 ;
415 ;
416 ;
417 ;
418 ;
419 ;
420 ;
421 ;
422 ;
423 ;
424 ;
425 ;
426 ;
427 ;
428 ;
429 ;
430 ;
431 ;
432 ;
433 ;
434 ;
435 ;
436 ;
437 ;
438 ;
439 ;
440 ;
441 ;
442 ;
443 ;
444 ;
445 ;
446 ;
447 ;
448 ;
449 ;
450 ;
451 ;
452 ;
453 ;
454 ;
455 ;
456 ;
457 ;
458 ;
459 ;
460 ;
461 ;
462 ;
463 ;
464 ;
465 ;
466 ;
467 ;
468 ;
469 ;
470 ;
471 ;
472 ;
473 ;
474 ;
475 ;
476 ;
477 ;
478 ;
479 ;
480 ;
481 ;
482 ;
483 ;
484 ;
485 ;
486 ;
487 ;
488 ;
489 ;
490 ;
491 ;
492 ;
493 ;
494 ;
495 ;
496 ;
497 ;
498 ;
499 ;
500 ;
501 ;
502 ;
503 ;
504 ;
505 ;
506 ;
507 ;
508 ;
509 ;
510 ;
511 ;
512 ;
513 ;
514 ;
515 ;
516 ;
517 ;
518 ;
519 ;
520 ;
521 ;
522 ;
523 ;
524 ;
525 ;
526 ;
527 ;
528 ;
529 ;
530 ;
531 ;
532 ;
533 ;
534 ;
535 ;
536 ;
537 ;
538 ;
539 ;
540 ;
541 ;
542 ;
543 ;
544 ;
545 ;
546 ;
547 ;
548 ;
549 ;
550 ;
551 ;
552 ;
553 ;
554 ;
555 ;
556 ;
557 ;
558 ;
559 ;
560 ;
561 ;
562 ;
563 ;
564 ;
565 ;
566 ;
567 ;
568 ;
569 ;
570 ;
571 ;
572 ;
573 ;
574 ;
575 ;
576 ;
577 ;
578 ;
579 ;
580 ;
581 ;
582 ;
583 ;
584 ;
585 ;
586 ;
587 ;
588 ;
589 ;
590 ;
591 ;
592 ;
593 ;
594 ;
595 ;
596 ;
597 ;
598 ;
599 ;
600 ;
601 ;
602 ;
603 ;
604 ;
605 ;
606 ;
607 ;
608 ;
609 ;
610 ;
611 ;
612 ;
613 ;
614 ;
615 ;
616 ;
617 ;
618 ;
619 ;
620 ;
621 ;
622 ;
623 ;
624 ;
625 ;
626 ;
627 ;
628 ;
629 ;
630 ;
631 ;
632 ;
633 ;
634 ;
635 ;
636 ;
637 ;
638 ;
639 ;
640 ;
641 ;
642 ;
643 ;
644 ;
645 ;
646 ;
647 ;
648 ;
649 ;
650 ;
651 ;
652 ;
653 ;
654 ;
655 ;
656 ;
657 ;
658 ;
659 ;
660 ;
661 ;
662 ;
663 ;
664 ;
665 ;
666 ;
667 ;
668 ;
669 ;
670 ;
671 ;
672 ;
673 ;
674 ;
675 ;
676 ;
677 ;
678 ;
679 ;
680 ;
681 ;
682 ;
683 ;
684 ;
685 ;
686 ;
687 ;
688 ;
689 ;
690 ;
691 ;
692 ;
693 ;
694 ;
695 ;
696 ;
697 ;
698 ;
699 ;
700 ;
701 ;
702 ;
703 ;
704 ;
705 ;
706 ;
707 ;
708 ;
709 ;
710 ;
711 ;
712 ;
713 ;
714 ;
715 ;
716 ;
717 ;
718 ;
719 ;
720 ;
721 ;
722 ;
723 ;
724 ;
725 ;
726 ;
727 ;
728 ;
729 ;
730 ;
731 ;
732 ;
733 ;
734 ;
735 ;
736 ;
737 ;
738 ;
739 ;
740 ;
741 ;
742 ;
743 ;
744 ;
745 ;
746 ;
747 ;
748 ;
749 ;
750 ;
751 ;
752 ;
753 ;
754 ;
755 ;
756 ;
757 ;
758 ;
759 ;
760 ;
761 ;
762 ;
763 ;
764 ;
765 ;
766 ;
767 ;
768 ;
769 ;
770 ;
771 ;
772 ;
773 ;
774 ;
775 ;
776 ;
777 ;
778 ;
779 ;
780 ;
781 ;
782 ;
783 ;
784 ;
785 ;
786 ;
787 ;
788 ;
789 ;
790 ;
791 ;
792 ;
793 ;
794 ;
795 ;
796 ;
797 ;
798 ;
799 ;
800 ;
801 ;
802 ;
803 ;
804 ;
805 ;
806 ;
807 ;
808 ;
809 ;
810 ;
811 ;
812 ;
813 ;
814 ;
815 ;
816 ;
817 ;
818 ;
819 ;
820 ;
821 ;
822 ;
823 ;
824 ;
825 ;
826 ;
827 ;
828 ;
829 ;
830 ;
831 ;
832 ;
833 ;
834 ;
835 ;
836 ;
837 ;
838 ;
839 ;
840 ;
841 ;
842 ;
843 ;
844 ;
845 ;
846 ;
847 ;
848 ;
849 ;
850 ;
851 ;
852 ;
853 ;
854 ;
855 ;
856 ;
857 ;
858 ;
859 ;
860 ;
861 ;
862 ;
863 ;
864 ;
865 ;
866 ;
867 ;
868 ;
869 ;
870 ;
871 ;
872 ;
873 ;
874 ;
875 ;
876 ;
877 ;
878 ;
879 ;
880 ;
881 ;
882 ;
883 ;
884 ;
885 ;
886 ;
887 ;
888 ;
889 ;
890 ;
891 ;
892 ;
893 ;
894 ;
895 ;
896 ;
897 ;
898 ;
899 ;
900 ;
901 ;
902 ;
903 ;
904 ;
905 ;
906 ;
907 ;
908 ;
909 ;
910 ;
911 ;
912 ;
913 ;
914 ;
915 ;
916 ;
917 ;
918 ;
919 ;
920 ;
921 ;
922 ;
923 ;
924 ;
925 ;
926 ;
927 ;
928 ;
929 ;
930 ;
931 ;
932 ;
933 ;
934 ;
935 ;
936 ;
937 ;
938 ;
939 ;
940 ;
941 ;
942 ;
943 ;
944 ;
945 ;
946 ;
947 ;
948 ;
949 ;
950 ;
951 ;
952 ;
953 ;
954 ;
955 ;
956 ;
957 ;
958 ;
959 ;
960 ;
961 ;
962 ;
963 ;
964 ;
965 ;
966 ;
967 ;
968 ;
969 ;
970 ;
971 ;
972 ;
973 ;
974 ;
975 ;
976 ;
977 ;
978 ;
979 ;
980 ;
981 ;
982 ;
983 ;
984 ;
985 ;
986 ;
987 ;
988 ;
989 ;
990 ;
991 ;
992 ;
993 ;
994 ;
995 ;
996 ;
997 ;
998 ;
999 ;
1000 ;

```

```

1 ;
2 ;
3 ;
4 ;
5 ;
6 ;
7 ;
8 ;
9 ;
10 ;
11 ;
12 ;
13 ;
14 ;
15 ;
16 ;
17 ;
18 ;
19 ;
20 ;
21 ;
22 ;
23 ;
24 ;
25 ;
26 ;
27 ;
28 ;
29 ;
30 ;
31 ;
32 ;
33 ;
34 ;
35 ;
36 ;
37 ;
38 ;
39 ;
40 ;
41 ;
42 ;
43 ;
44 ;
45 ;
46 ;
47 ;
48 ;
49 ;
50 ;
51 ;
52 ;
53 ;
54 ;
55 ;
56 ;
57 ;
58 ;
59 ;
60 ;
61 ;
62 ;
63 ;
64 ;
65 ;
66 ;
67 ;
68 ;
69 ;
70 ;
71 ;
72 ;
73 ;
74 ;
75 ;
76 ;
77 ;
78 ;
79 ;
80 ;
81 ;
82 ;
83 ;
84 ;
85 ;
86 ;
87 ;
88 ;
89 ;
90 ;
91 ;
92 ;
93 ;
94 ;
95 ;
96 ;
97 ;
98 ;
99 ;
100 ;
101 ;
102 ;
103 ;
104 ;
105 ;
106 ;
107 ;
108 ;
109 ;
110 ;
111 ;
112 ;
113 ;
114 ;
115 ;
116 ;
117 ;
118 ;
119 ;
120 ;
121 ;
122 ;
123 ;
124 ;
125 ;
126 ;
127 ;
128 ;
129 ;
130 ;
131 ;
132 ;
133 ;
134 ;
135 ;
136 ;
137 ;
138 ;
139 ;
140 ;
141 ;
142 ;
143 ;
144 ;
145 ;
146 ;
147 ;
148 ;
149 ;
150 ;
151 ;
152 ;
153 ;
154 ;
155 ;
156 ;
157 ;
158 ;
159 ;
160 ;
161 ;
162 ;
163 ;
164 ;
165 ;
166 ;
167 ;
168 ;
169 ;
170 ;
171 ;
172 ;
173 ;
174 ;
175 ;
176 ;
177 ;
178 ;
179 ;
180 ;
181 ;
182 ;
183 ;
184 ;
185 ;
186 ;
187 ;
188 ;
189 ;
190 ;
191 ;
192 ;
193 ;
194 ;
195 ;
196 ;
197 ;
198 ;
199 ;
200 ;
201 ;
202 ;
203 ;
204 ;
205 ;
206 ;
207 ;
208 ;
209 ;
210 ;
211 ;
212 ;
213 ;
214 ;
215 ;
216 ;
217 ;
218 ;
219 ;
220 ;
221 ;
222 ;
223 ;
224 ;
225 ;
226 ;
227 ;
228 ;
229 ;
230 ;
231 ;
232 ;
233 ;
234 ;
235 ;
236 ;
237 ;
238 ;
239 ;
240 ;
241 ;
242 ;
243 ;
244 ;
245 ;
246 ;
247 ;
248 ;
249 ;
250 ;
251 ;
252 ;
253 ;
254 ;
255 ;
256 ;
257 ;
258 ;
259 ;
260 ;
261 ;
262 ;
263 ;
264 ;
265 ;
266 ;
267 ;
268 ;
269 ;
270 ;
271 ;
272 ;
273 ;
274 ;
275 ;
276 ;
277 ;
278 ;
279 ;
280 ;
281 ;
282 ;
283 ;
284 ;
285 ;
286 ;
287 ;
288 ;
289 ;
290 ;
291 ;
292 ;
293 ;
294 ;
295 ;
296 ;
297 ;
298 ;
299 ;
300 ;
301 ;
302 ;
303 ;
304 ;
305 ;
306 ;
307 ;
308 ;
309 ;
310 ;
311 ;
312 ;
313 ;
314 ;
315 ;
316 ;
317 ;
318 ;
319 ;
320 ;
321 ;
322 ;
323 ;
324 ;
325 ;
326 ;
327 ;
328 ;
329 ;
330 ;
331 ;
332 ;
333 ;
334 ;
335 ;
336 ;
337 ;
338 ;
339 ;
340 ;
341 ;
342 ;
343 ;
344 ;
345 ;
346 ;
347 ;
348 ;
349 ;
350 ;
351 ;
352 ;
353 ;
354 ;
355 ;
356 ;
357 ;
358 ;
359 ;
360 ;
361 ;
362 ;
363 ;
364 ;
365 ;
366 ;
367 ;
368 ;
369 ;
370 ;
371 ;
372 ;
373 ;
374 ;
375 ;
376 ;
377 ;
378 ;
379 ;
380 ;
381 ;
382 ;
383 ;
384 ;
385 ;
386 ;
387 ;
388 ;
389 ;
390 ;
391 ;
392 ;
393 ;
394 ;
395 ;
396 ;
397 ;
398 ;
399 ;
400 ;
401 ;
402 ;
403 ;
404 ;
405 ;
406 ;
407 ;
408 ;
409 ;
410 ;
411 ;
412 ;
413 ;
414 ;
415 ;
416 ;
417 ;
418 ;
419 ;
420 ;
421 ;
422 ;
423 ;
424 ;
425 ;
426 ;
427 ;
428 ;
429 ;
430 ;
431 ;
432 ;
433 ;
434 ;
435 ;
436 ;
437 ;
438 ;
439 ;
440 ;
441 ;
442 ;
443 ;
444 ;
445 ;
446 ;
447 ;
448 ;
449 ;
450 ;
451 ;
452 ;
453 ;
454 ;
455 ;
456 ;
457 ;
458 ;
459 ;
460 ;
461 ;
462 ;
463 ;
464 ;
465 ;
466 ;
467 ;
468 ;
469 ;
470 ;
471 ;
472 ;
473 ;
474 ;
475 ;
476 ;
477 ;
478 ;
479 ;
480 ;
481 ;
482 ;
483 ;
484 ;
485 ;
486 ;
487 ;
488 ;
489 ;
490 ;
491 ;
492 ;
493 ;
494 ;
495 ;
496 ;
497 ;
498 ;
499 ;
500 ;
501 ;
502 ;
503 ;
504 ;
505 ;
506 ;
507 ;
508 ;
509 ;
510 ;
511 ;
512 ;
513 ;
514 ;
515 ;
516 ;
517 ;
518 ;
519 ;
520 ;
521 ;
522 ;
523 ;
524 ;
525 ;
526 ;
527 ;
528 ;
529 ;
530 ;
531 ;
532 ;
533 ;
534 ;
535 ;
536 ;
537 ;
538 ;
539 ;
540 ;
541 ;
542 ;
543 ;
544 ;
545 ;
546 ;
547 ;
548 ;
549 ;
550 ;
551 ;
552 ;
553 ;
554 ;
555 ;
556 ;
557 ;
558 ;
559 ;
560 ;
561 ;
562 ;
563 ;
564 ;
565 ;
566 ;
567 ;
568 ;
569 ;
570 ;
571 ;
572 ;
573 ;
574 ;
575 ;
576 ;
577 ;
578 ;
579 ;
580 ;
581 ;
582 ;
583 ;
584 ;
585 ;
586 ;
587 ;
588 ;
589 ;
590 ;
591 ;
592 ;
593 ;
594 ;
595 ;
596 ;
597 ;
598 ;
599 ;
600 ;
601 ;
602 ;
603 ;
604 ;
605 ;
606 ;
607 ;
608 ;
609 ;
610 ;
611 ;
612 ;
613 ;
614 ;
615 ;
616 ;
617 ;
618 ;
619 ;
620 ;
621 ;
622 ;
623 ;
624 ;
625 ;
626 ;
627 ;
628 ;
629 ;
630 ;
631 ;
632 ;
633 ;
634 ;
635 ;
636 ;
637 ;
638 ;
639 ;
640 ;
641 ;
642 ;
643 ;
644 ;
645 ;
646 ;
647 ;
648 ;
649 ;
650 ;
651 ;
652 ;
653 ;
654 ;
655 ;
656 ;
657 ;
658 ;
659 ;
660 ;
661 ;
662 ;
663 ;
664 ;
665 ;
666 ;
667 ;
668 ;
669 ;
670 ;
671 ;
672 ;
673 ;
674 ;
675 ;
676 ;
677 ;
678 ;
679 ;
680 ;
681 ;
682 ;
683 ;
684 ;
685 ;
686 ;
687 ;
688 ;
689 ;
690 ;
691 ;
692 ;
693 ;
694 ;
695 ;
696 ;
697 ;
698 ;
699 ;
700 ;
701 ;
702 ;
703 ;
704 ;
705 ;
706 ;
707 ;
708 ;
709 ;
710 ;
711 ;
712 ;
713 ;
714 ;
715 ;
716 ;
717 ;
718 ;
719 ;
720 ;
721 ;
722 ;
723 ;
724 ;
725 ;
726 ;
727 ;
728 ;
729 ;
730 ;
731 ;
732 ;
733 ;
734 ;
735 ;
736 ;
737 ;
738 ;
739 ;
740 ;
741 ;
742 ;
743 ;
744 ;
745 ;
746 ;
747 ;
748 ;
749 ;
750 ;
751 ;
752 ;
753 ;
754 ;
755 ;
756 ;
757 ;
758 ;
759 ;
760 ;
761 ;
762 ;
763 ;
764 ;
765 ;
766 ;
767 ;
768 ;
769 ;
770 ;
771 ;
772 ;
773 ;
774 ;
775 ;
776 ;
777 ;
778 ;
779 ;
780 ;
781 ;
782 ;
783 ;
784 ;
785 ;
786 ;
787 ;
788 ;
789 ;
790 ;
791 ;
792 ;
793 ;
794 ;
795 ;
796 ;
797 ;
798 ;
799 ;
800 ;
801 ;
802 ;
803 ;
804 ;
805 ;
806 ;
807 ;
808 ;
809 ;
810 ;
811 ;
812 ;
813 ;
814 ;
815 ;
816 ;
817 ;
818 ;
819 ;
820 ;
821 ;
822 ;
823 ;
824 ;
825 ;
826 ;
827 ;
828 ;
829 ;
830 ;
831 ;
832 ;
833 ;
834 ;
835 ;
836 ;
837 ;
838 ;
839 ;
840 ;
841 ;
842 ;
843 ;
844 ;
845 ;
846 ;
847 ;
848 ;
849 ;
850 ;
851 ;
852 ;
853 ;
854 ;
855 ;
856 ;
857 ;
858 ;
859 ;
860 ;
861 ;
862 ;
863 ;
864 ;
865 ;
866 ;
867 ;
868 ;
869 ;
870 ;
871 ;
872 ;
873 ;
874 ;
875 ;
876 ;
877 ;
878 ;
879 ;
880 ;
881 ;
882 ;
883 ;
884 ;
885 ;
886 ;
887 ;
888 ;
889 ;
890 ;
891 ;
892 ;
893 ;
894 ;
895 ;
896 ;
897 ;
898 ;
899 ;
900 ;
901 ;
902 ;
903 ;
904 ;
905 ;
906 ;
907 ;
908 ;
909 ;
910 ;
911 ;
912 ;
913 ;
914 ;
915 ;
916 ;
917 ;
918 ;
919 ;
920 ;
921 ;
922 ;
923 ;
924 ;
925 ;
926 ;
927 ;
928 ;
929 ;
930 ;
931 ;
932 ;
933 ;
934 ;
935 ;
936 ;
937 ;
938 ;
939 ;
940 ;
941 ;
942 ;
943 ;
944 ;
945 ;
946 ;
947 ;
948 ;
949 ;
950 ;
951 ;
952 ;
953 ;
954 ;
955 ;
956 ;
957 ;
958 ;
959 ;
960 ;
961 ;
962 ;
963 ;
964 ;
965 ;
966 ;
967 ;
968 ;
969 ;
970 ;
971 ;
972 ;
973 ;
974 ;
975 ;
976 ;
977 ;
978 ;
979 ;
980 ;
981 ;
982 ;
983 ;
984 ;
985 ;
986 ;
987 ;
988 ;
989 ;
990 ;
991 ;
992 ;
993 ;
994 ;
995 ;
996 ;
997 ;
998 ;
999 ;
1000 ;

```

```

175 ;
176 ; STRUCTS
177 ;
178
179 (defmethod ako ((s symbol) kind) (ako (symbol-name s) kind))
180 (defmethod ako ((s string) kind)
181   (let
182     ((l1 '(:ignore #\:) (klass #\!) (less #\~) (more #\+) (goal #\+ #\~ #\!)))
183     (l2 '(num #\$))))
184   (or (member (char s (1- (length s))) (cdr (assoc kind 1)))
185       (member (char s 0) (cdr (assoc kind 12))))))
186 ;
187 ;
188 ;
189 ; check for certain 'kind's or suffixes or prefixes
190 (defstruct sym (:constructor %make-sym) (n 0) at name all mode (most 0))
191
192 (defun make-sym (&optional (at 0) (name ""))
193   (%make-sym :at at :name name))
194
195 (defmethod add ((self sym) x)
196   (with-slots (n all mode most) self
197     (unless (eq x #\?)
198       (incf n)
199       (let ((now (incf (has x all))))
200         (if (> now most)
201             (setf most now
202                 mode x))))))
203   x)
204
205 (defmethod div ((self sym)) (ent (sym-all self)))
206 (defmethod mid ((self sym)) (sym-mode self))
207 ;
208 ;
209 (defstruct num (:constructor %make-num) (n 0) at name
210   (all (make-array 5 :fill-pointer 0))
211   (size (? enough))
212   ok w (hi -1E32) (lo 1E32))
213
214 (defun make-num (&optional (at 0) (name ""))
215   (%make-num :at at :name name :w (if (ako name 'less) -1 1)))
216
217 (defmethod add ((self num) x)
218   (with-slots (n lo hi ok all size) self
219     (unless (eq x #\?)
220       (incf n)
221       (setf lo (min x lo)
222             hi (max x hi))
223       (cond ((< (length all) size) (vector-push-extend x all) (setf ok nil))
224             ((< (randf) (/ size n)) (setf (elt all (randi (length all))) x
225                                           ok nil))))))
226   x)
227
228 (defmethod div ((self num)) (sd (holds self)))
229 (defmethod mid ((self num)) (per (holds self)))
230
231 (defmethod holds ((self num))
232   (with-slots (ok all) self
233     (unless ok (setf all (sort all #'<)))
234     (setf ok t)
235     all))
236 ;
237 ;
238 (defstruct cols (:constructor %make-cols) all x y klass)
239
240 (defun make-cols (names &aux (at -1) x y klass all)
241   (dolist (name names (%make-cols :all (reverse all) :x x :y y :klass klass))
242     (let* ((what (if (ako name 'num) #'make-num #'make-sym))
243            (now (funcall what (incf at) name)))
244       (push now all)
245       (when (not (ako name 'ignore))
246         (if (ako name 'goal) (push now x) (push now y))
247         (if (ako name 'klass) (setf klass now))))))
248
249 ;
250 ;
251 ;
252 (defstruct eggs (:constructor %make-egs) rows cols)
253
254 (defun make-egs (&optional from)
255   (let ((self (%make-egs)))
256     (cond ((consp from)
257           (dolist (row from) (add self row)))
258           ((stringp from)
259            (with-csv (row (? files))
260              (add self (mapcar #'thing (str2list row))))))
261           (self)))
262
263 (defmethod add ((self eggs) row)
264   (with-slots (cols rows) self
265     (if cols
266         (push (mapcar #'add cols row) rows)
267         (setf cols (make-cols row))))
268   row)

```

```

269 ;
270 ; UNIT TESTS
271 ;
272
273 (deftest go.cells () (print (mapcar #'thing (str2cells "23,asda,34.1"))))
274
275 (deftest go.has ()
276   (let (x y)
277     (incf (has 'aa x))
278     (incf (has 'aa x))
279     (print x)
280     (ok (eq 2 (cdr (assoc 'aa x))) "inc assoc list")))
281
282 (deftest go.csv (&aux (n 0))
283   (with-csv (row (? file)) (incf n))
284   (ok (eq 399 n) "reading lines")))
285
286 (deftest go.normal ()
287   (dolist (n '(10000 5000 2500 1250 500 250 125 60 30 15))
288     (let (l)
289       (setf l (dotimes (i n (sort l #'<)) (push (normal) l)))
290       (format t "~5@A:~6,4f:~6,4f~%" n (sd l) (per l)))))
291
292 (deftest go.rand (&aux l)
293   (dotimes (i 50) (push (randi 4) l))
294   (print (sort l #'<)))
295
296 (deftest go.ent ()
297   (let (x)
298     (incf (has 'this x) 4)
299     (incf (has 'that x) 2)
300     (incf (has 'other x) 1)
301     (ok (<= 1.378 (ent x) 1.379) "diversity")))
302
303 (deftest go.num (&aux (num (make-num)))
304   (dotimes (i 100000 (print (holds num))) (add num i)))
305
306 (deftest go.sym (&aux (sym (make-sym)))
307   (dotimes (i 100000 (print (sym-all sym))) (add sym (randi 10)))))
308
309 (deftest go.cols (&aux c)
310   (setf c (make-cols '("$ss" "age!" "$weight-")))
311   (print c))
312
313 (deftest go.egs (&aux e)
314   (make-egs (? file)))
315
316 ;;;;-----
317 (main)

```