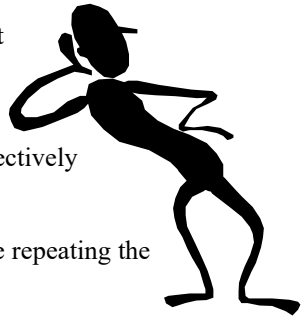
PROGRAM 482j

(Insertion Sort)

Program Description: Read a series of two-digit integers into an array from an unordered external file. Write a function that accepts that array, sorts the integers by using a INSERTION SORT and returns the ordered array to the main program for output.

It insertion sort rests on the assumption that:



1. At the outset, when the list is considered to consist of one element only, it is already in its correct order.

2. Now the list up to the second element is considered. If the second element is smaller than the first, a swap is made. This effectively "inserts" the second element into its correct order.

3. The list is now extended to the third element, and so on, each time repeating the process of migrating the item into its correct position.

An analogy for this sorting algorithm is a card player who picks up his cards one at a time and arranges (inserts) them into the correct order.

Example:

Original List:	12	5	8	3	1	1
Pass 1	12					
Pass 2	5	12				
Pass 3	5	8	12			
Pass 4	3	5	8	12		
Pass 5	1	3	5	8	12	
Pass 6	1	1	3	5	8	12
Sort completed						

The insertion sort algorithm is, on the average, about twice as fast as the selection sort (and four times faster than the bubble sort).

Statements Required: input, output, decision making, loop control, array

Data Location: numsort.dat

Exact copy: (see next page)

Insertion Sort

Original List:

388 206 868 857 239 659 687 504 426 132 854 734 37 299 957 920 734 179 13
208 953 568 374 144 159 545 228 417 513 148 477 821 333 210 846 207 654 580
615 754 740 296 580 647 631 133 666 167 931 943 242 413 220 955 669 515 75
401 63 817 476 303 625 171 482 531 968 616 85 186 707 92 692 813 278 440
230 181 596 16 253 484 268 37 144 676 869 92 436 310 575 5 833 2 449
5 868 276 820 338 783 438 717 463 606 258 278 57 881 151 216 111 399 345
107 210 857 695 462 495 368 538 813 456 386 200 584 513 544 353 664 340 763
27 684 167 938 998 186 416 962 19 780 94 560 826 133 207 64 581 487 628
550 432 999 414 49 3 879 58 578 659 554 231 790 6 34 568 880 590 975
379 460 793 349 354 485 205 620 13 379 340 366 559 513 909 321 234 235 376
582 490 740 970 738 635 238 540 274 872 689 995 945 699 796 223 717 408 66
830 772 514 971 357 511 87 826 54 326 458 835 149 730 401 51 442 536 631
156 359 511 355 399 569 353 43 654 559 919 719 841 635 109 704 927 501 950
527 79 401 518 792 121 415 17 827 751 768 263 820 910 698 564 585 113 2
203 190 339 866 707 539 362 836 436 118 79 719 57 356 209 659 755 616 252
448 617 162 217 109 999 914 165 882 858 2 33 366 778 736 684 583 183 159
529 239 311 829 398 799 330 591 511 308 251 305 306 830 687 11 752 685 920
639 858 888 729 583 45 580 329 618 319 52 921 761 572 209 418 382 116 480
831 374 571 822 203 134 225 692 461 800 666 587 672 3 446 178 394 853 446
13 717 480 1 156 350 115 55 233 392 156 523 614 502 355 129 754 204 304
222 308 13 786 451 121 585 538 642 457 763 430 581 561 717 749 245 13 588
470 736 321 526 113 785 717 393 25 585 974 574 678 229 420 537 220 519 17
799 6 837 989 861 754 164 191 316 301 95 898 678 97 165 95 405 99 802
97 424 289 145 944 456 173 604 759 416 567 877 960 60 81 399 215 934 335
269 418 534 511 155 396 271 155 203 919 323 358 407 558 795 934 125 27 707
534 827 168 583 655 396 207 930 966 490 442 669 322 555 753 798 634 690 399
90 832 981 51 235 599

Sorted List: 1 2 2 2 3 3 5 5 6 6 11 13 13 13 13 16 17
17 19 25 27 27 33 34 37 37 43 45 49 51 51 52 54 55 57 57
58 60 63 64 66 75 79 79 81 85 87 90 92 92 94 95 95 97 97
99 107 109 109 111 113 113 115 116 118 121 121 125 129 132 133 133 133 134
144 144 145 148 149 151 155 155 156 156 156 159 159 162 164 165 165 167 167
168 171 173 178 179 181 183 186 186 190 191 200 203 203 203 204 205 206 207
207 207 208 209 209 210 210 215 216 217 220 220 222 223 225 228 229 230 231
233 234 235 235 238 239 239 242 245 251 252 253 258 263 268 269 271 274 276
278 278 289 296 299 301 303 304 305 306 308 308 310 311 316 319 321 321 322
323 326 329 330 333 335 338 339 340 340 345 349 350 353 353 354 355 355 356
357 358 359 362 366 366 368 374 374 376 379 379 382 386 388 392 393 394 396
396 398 399 399 399 399 401 401 401 405 407 408 413 414 415 416 416 417 418
418 420 424 426 430 432 436 436 438 440 442 442 446 446 448 449 451 456 456
457 458 460 461 462 463 470 476 477 480 480 482 484 485 487 490 490 495 501
502 504 511 511 511 511 513 513 513 514 515 518 519 523 526 527 529 531 534
534 536 537 538 538 539 540 544 545 550 554 555 558 559 559 560 561 564 567
568 568 569 571 572 574 575 578 580 580 580 581 581 582 583 583 583 584 585
585 585 587 588 590 591 596 599 604 606 614 615 616 616 617 618 620 625 628
631 631 634 635 635 639 642 647 654 654 655 659 659 659 664 666 666 669 669
672 676 678 678 684 684 685 687 687 689 690 692 692 695 698 699 704 707 707
707 717 717 717 717 717 719 719 729 730 734 734 736 736 738 740 740 749 751
752 753 754 754 754 755 759 761 763 763 768 772 778 780 783 785 786 790 792
793 795 796 798 799 799 800 802 813 813 817 820 820 821 822 826 826 827 827
829 830 830 831 832 833 835 836 837 841 846 853 854 857 857 858 858 861 866
868 868 869 872 877 879 880 881 882 888 898 909 910 914 919 919 920 920 921
927 930 931 934 934 938 943 944 945 950 953 955 957 960 962 966 968 970 971
974 975 981 989 995 998 999 999