31. If a Java 5.0 generic method is called with three different generic parameters, how many versions of the method will be generated by the compiler?
32. What are the design issues for functions?
33. What two languages allow multiple values to be returned from a function?
34. What languages allow the user to overload operators?
35. In what ways are coroutines different from conventional subprograms?

PROBLEM SET

1. What are arguments for and against a user program building additional definitions for existing operators, as can be done in Ada and C++? Do you believe such user-defined operator overloading is good or bad? Support your answer.
2. In most Fortran IV implementations, parameters were passed by reference, using access path transmission only. State both the advantages and disadvantages of this design choice.
3. Argue in support of the Ada 83 designers' decision to allow the implementor to choose between implementing **in out** mode parameters by copy or by reference.
4. Suppose you wish to write a method that prints a heading on a new output page, along with a page number that is 1 in the first activation and that increases by 1 with each subsequent activation. Can this be done without parameters and without reference to nonlocal variables in Java? Can it be done in C#?
5. Consider the following program written in C syntax:

```
void swap(int a, int b) {
  int temp;
  temp = a;
  a = b;
  b = temp;
}
void main() {
  int value = 2, list[5] = {1, 3, 5, 7, 9};
  swap(value, list[0]);
  swap(list[0], list[1]);
  swap(value, list[value]);
}
```

For each of the following parameter-passing methods, what are all of the values of the variables `value` and `list` after each of the three calls to `swap`?

a. Passed by value
b. Passed by reference
c. Passed by value-result

6. Present one argument against providing both static and dynamic local variables in subprograms.

7. Consider the following program written in C syntax:

```
void fun (int first, int second) {
  first += first;
  second += second;
}
void main() {
  int list[2] = {1, 3};
  fun(list[0], list[1]);
}
```

For each of the following parameter-passing methods, what are the values of the `list` array after execution?

a. Passed by value
b. Passed by reference
c. Passed by value-result

8. Argue against the C design of providing only function subprograms.

9. From a textbook on Fortran, learn the syntax and semantics of statement functions. Justify their existence in Fortran.

10. Study the methods of user-defined operator overloading in C++ and Ada, and write a report comparing the two using our criteria for evaluating languages.

11. C# supports out-mode parameters, but neither Java nor C++ does. Give an explanation of this difference.

12. Research Jensen's Device, which was a widely known use of pass-by-name parameters, and write a short description of what it is and how it can be used.

13. Study the iterator mechanisms of Ruby and CLU and list their similarities and differences.

14. Speculate on the issue of allowing nested subprograms in programming languages—why are they not allowed in many contemporary languages?

15. What are at least two arguments against the use of pass-by-name parameters?

16. Write a detailed comparison of the generic subprograms of Java 5.0 and C# 2005.