p394
MR

1 Comparative Programming languages 2004

(a) Discuss to what extent a programmer can expect a program that conforms to a standard to generate identical results when run under different comforming compilers on different machines. [6 marks]

(b) ALGOL 60 provided call by value and call by name, Pascal provided call by value and call by reference, and ALGOL-W provided a variety of calling methods including call by result and call by value-result. Briefly describe the calling mechanism just mentioned and discuss why most modern programming languages only provide call by value. [8 marks]

(c) Discuss the reasons why languages such as Fortran, Algol and PL/I designed in 1950s and 1960s are now less popular than the more modern languages designed in the last 20 years. [6 marks]

ANSWER NOTES:

(a) Points to mention:

Different M/Cs have different word lengths, different character codes, different representations for floating point number and different instruction sets. To allow reasonable efficiency language standards typically allow machine dependent representations to be used. They also typically allow some choice in the order of evaluation of operands in expressions, as in f(x)+g(y). If one of these calls has side effects it may affect the value of the other operand. Sometimes programs genuinely wish to produce different results on different run as in RSA security applications. In the 1960s and 70s mathines were too small to allow compilers to do much optimisation. Nowaday language specifications could be stricter allowing compilers to optimise code only when such changes could be proved to leave the result unchanged.

(b) These calling method are all bookwork and described in the lectures. The other two modes of calling for Algol-W are call by name and call by value which are the same as for Algol 60. They need not be mentioned to obtain full marks. Modern languages do not need these calling mechanisms since there effects can be achieved easily in other ways, by for instance passing pointers, functions or object as arguments.

(c) These early languages were designed before there was much experience in the design of programming languages. To compete they had to be efficient and easy to compile with small compilers. They were typically quite primitive and inconvenient to use. They, particularly Fortran and Algol, lacked many of the features available in modern languages. Fortran and Algol has no data structure mechanism, no pointers and Fortran originally did not allow recursion. Exception mechanisms were either non-existant or primitive and the could not handle higher order functions or object orientation. Popular languages gain more investment in good compilers, program development system, documentation, and have a larger population of programmers expert in their use. Other languages tend to wither and die.