ACM Turing Award Winners from the PL/Compilers Area
• **Alan Perlis** (1966, first ever Turing Award) - For his influence in the area of advanced programming techniques and compiler construction

• **Edsger Dijkstra** (1972) - For fundamental contributions to programming as a high, intellectual challenge; for eloquent insistence and practical demonstration that programs should be composed correctly, not just debugged into correctness; for illuminating perception of problems at the foundations of program design

• **Donald Knuth** (1974) - For his major contributions to the analysis of algorithms and the design of programming languages, and in particular for his contributions to the "art of computer programming" through his well-known books in a continuous series by this title.

• **John Backus** (1977) - For profound, influential, and lasting contributions to the design of practical high-level programming systems, notably through his work on FORTRAN, and for seminal publication of formal procedures for the specification of programming languages.

• **Robert Floyd** (1978) - For having a clear influence on methodologies for the creation of efficient and reliable software, and for helping to founded the following important subfields of computer science: the theory of parsing, the semantics of programming languages, automatic program verification, automatic program synthesis, and analysis of algorithms.
• **Ken Iverson** (1979) - For his pioneering effort in programming languages and mathematical notation resulting in what the computing field now knows as APL, for his contributions to the implementation of interactive systems, to educational uses of APL, and to programming language theory and practice.

• **Tony Hoare** (1980) - For his fundamental contributions to the definition and design of programming languages.

• **Niklaus Wirth** (1984) - For developing a sequence of innovative computer languages, EULER, ALGOL-W, MODULA and PASCAL. PASCAL has become pedagogically significant and has provided a foundation for future computer language, systems, and architectural research.

• **John Cocke** (1987) – For significant contributions in the design and theory of compilers, the architecture of large systems and the development of reduced instruction set computers (RISC); for discovering and systematizing many fundamental transformations now used in optimizing compilers including reduction of operator strength, elimination of common subexpressions, register allocation, constant propagation, and dead code elimination.
• **Robin Milner** (1991) – For three distinct and complete achievements: LCF, the mechanization of Scott's Logic of Computable Functions, probably the first theoretically based yet practical tool for machine assisted proof construction; ML, the first language to include polymorphic type inference together with a type-safe exception-handling mechanism; CCS, a general theory of concurrency. In addition, he formulated and strongly advanced full abstraction, the study of the relationship between operational and denotational semantics.

• **Ole-Johan Dahl and Kristen Nygaard** (2001) - for ideas fundamental to the emergence of object-oriented programming, through their design of the programming languages Simula I and Simula 67.

• **Alan Key** (2003) - For pioneering many of the ideas at the root of contemporary object-oriented programming languages, leading the team that developed Smalltalk, and for fundamental contributions to personal computing.

• **Peter Naur** (2005) - For fundamental contributions to programming language design and the definition of Algol 60, to compiler design, and to the art and practice of computer programming.
• **Fran Allen** (2006) - For pioneering contributions to the theory and practice of optimizing compiler techniques that laid the foundation for modern optimizing compilers and automatic parallel execution.

• **Barbara Liskov** (2008) - For contributions to practical and theoretical foundations of programming language and system design, especially related to data abstraction, fault tolerance, and distributed computing.

• **Alfred Aho and Jeffey Ullman** (2020) - For fundamental algorithms and theory underlying programming language implementation and for synthesizing these results and those of others in their highly influential books, which educated generations of computer scientists.