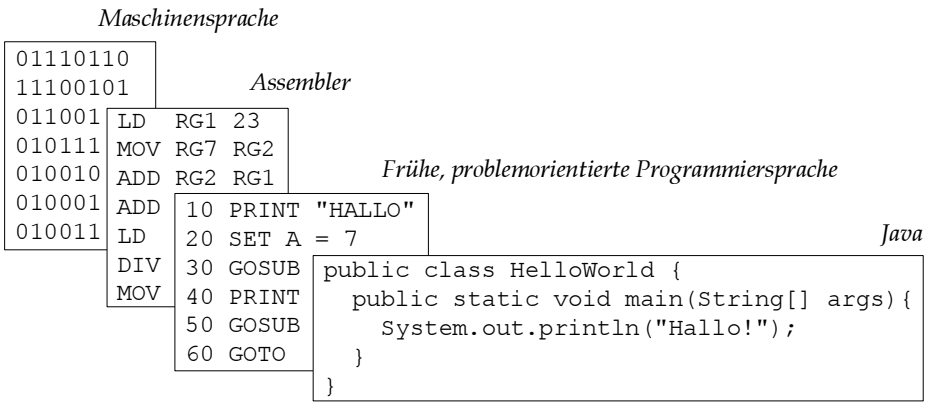

Abbildung 2.2: Programmiersprachen im Vergleich

Die Computer- und Software-Technik nahm seit ihren Kindertagen eine rasante Entwicklung von den Rechnern mit Transistortechnik und integrierten Schaltungen in Großrechnern und Minicomputern über die ersten Mikroprozessoren und Personal Computer Ende der siebziger Jahre bis zu den heutigen leistungsfähigen PCs, Workstations und Supercomputern. So ist es auch nicht verwunderlich, dass sich recht bald eine (seit 1960) eigenständige Wissenschaftsdisziplin entwickelte, die **Informatik**. Sie beschäftigt sich mit der theoretischen Analyse und Konzeption, aber auch mit der konkreten Realisierung von Computersystemen in den Bereichen Hardware, Software, Organisationsstrukturen und Anwender.

2.2 Was heißt Programmieren?

Wie bereits erwähnt, ist ein Programm nichts anderes als ein in einer **Programmiersprache** formulierter Algorithmus. Diese Sprache erlaubt es den Anwendern (Programmierern), mit dem Computer zu „sprechen“ und ihm Anweisungen zu geben. Dieses „Sprechen“ kann nun auf unterschiedliche Arten erfolgen. Man kann zum Beispiel eine Sprache verwenden, die der Computer (genau genommen der Prozessor) direkt „versteht“. Man nennt sie **Maschinensprache**. Da die Notation mit Nullen und Einsen nur schwer lesbar ist, verwendet man zur Formulierung von maschinennahen Programmen meist eine **Assemblersprache**, in der jedem binären Maschinencode ein entsprechender, aus Buchstaben und Ziffern bestehender Assemblercode entspricht. Programme in solchen Sprachen kann der Prozessor direkt ausführen, doch ist man dabei sehr vom Prozessortyp abhängig. Alternativ kann auch eine benutzernahe bzw. problemnahe Programmiersprache zum Einsatz kommen, die man als **höhere Programmiersprache** oder auch **problemorientierte Programmiersprache** bezeichnet. In diesem Fall benötigt man allerdings einen **Übersetzer** (in Form einer speziellen Software), der die Sätze der