Versuche, das Rechnen zu mechanisieren, sind bis an die Anfänge der Mathematik zurück feststellbar. Sie scheiterten aber meist an der technischen Realisierbarkeit. Erst im späteren Mittelalter tauchten die ersten brauchbaren rechnerischen Hilfsapparate auf.

ca 3000 aD Rechnen im Zweistromland:
Bekannt sind nur Addition und Subtraktion. Multiplikation wird mittels fortgesetzter binärer Addition ausgeführt.

aD Vor Christi Geburt wurde der ABAKUS als Rechenmaschine der Antike benutzt. Dieses Hilfsmittel wird bis in die heutige Zeit gebraucht. Dabei bedeutet die Stellung von Steinen, Perlen zum Mittelbalken eine Zahl. Mit dem Abakus kann man alle vier Grundrechenoperationen ausführen (+ - * /).

ca 800 Einführung der Zahl Null

1623 Geheimcode des Sir Francis Bacon
Sir Francis Bacon hat jedem Buchstaben des Alphabetes eine Kombination von insgesamt fünf „a" und „b" zugeordnet.

```
A     B     C     D     E     F
aaaaa aaaab aaaba aaabb aabaa aabab

G     H     I     K     L     M
aabba aabbb abaaa abaab ababa ababb

N     O     P     Q     R     S
abbaa abbab abbba abbbb baaaa baaab

T     U     V     W     X     Y
baaba baabb babaa babab babba babbb

Z
bbaaa
```

Was Sir Francis Bacon bei der Festlegung seines Codes vermutlich nicht realisierte, ist die Tatsache, dass er damit das erste Beispiel eines fünfstelligen Binärcodes beschrieb, ohne den die moderne Informatik undenkbar wäre. Binär heisst der Code deshalb, weil er nur zwei Symbole verwendet, a und b.

1623 Willhelm Schickard erfand eine Maschine, die alle vier Grundrechenoperationen ausführen konnte. Addition und Subtraktion mit Hilfe von Zahnrädern, Multiplikation und Division nach dem Prinzip des Rechenschiebers.

1642 Blaise Pascal stellte der Oeffentlichkeit eine Rechenmaschine vor, die addieren konnte. Eine Subtraktion wurde durch Komplementbildung und Addition ausgeführt.

1672 Wilhelm Leibniz entwickelte eine Maschine, die alle vier Grundoperationen mit Hilfe von Staffelwalzen ausführen konnte. Sie war der Prototyp aller späteren Handrechenmaschinen. Leibniz gilt auch als Erfinder des Dual- oder Binaersystemes.

1833 Charles Babbage entwarf Pläne für einen Rechenautomaten mit Speicher und externer Operationssteuerung. Diese Maschine (Analytical Engine) konnte beim damaligen Stand der Technik nicht realisiert werden.

1886 Hermann Hollerith entwickelte eine Sortier- und Zählmaschine. Hollerith gilt als Begründer der modernen Lochkartentechnik. Er arbeitete 1880 - 1882 für die amerikanische Volkszählung (500 Helfer, 7 Jahre sortieren und zählen).

Unter seiner Leitung folgten weitere Entwicklungen auf dem Gebiet Lochkarten - Technik (vor allem für statistische Auswertungen).

1941 Konrad Zuse konstruierte den ersten elektromechanischen Rechenautomaten. Rechenwerk und Speicher wurden aus zweiwertigen Elementen (Relais) aufgebaut. Aus diesem Grunde mussten alle Zahlen und Operationsbefehle binär verschlüsselt werden. Zuse verwendete ein externes Programm, das auf einem alten 16 mm Film gelocht worden war.

1944 Howard Aiken baute in den USA den ersten extern Programm - gesteuerten, elektromechanischen Rechenautomaten (Mark I).

1945 Die Amerikaner Eckert und Mauchly stellten den ersten elektronischen Rechenautomaten her, indem sie die Relais durch Elektronenröhren ersetzten. Der Rechner hatte 18‘000 Röhren. Er brauchte für eine Multiplikation nur 2.8 msec.

1948 John von Neumann formulierte das Konzept des speicherprogrammierten Automaten (Internes Programm). Er schuf damit den Prototyp des modernen Computers.
1. Computergeneration

1955 Transistoren und Dioden lösten die Elektronenröhre ab.
2. Computergeneration

1963 Erstmals wurden integrierte Schaltkreise mit relativ geringem Integrationsgrad eingebaut.
3. Computergeneration

1970 Kamen die ersten Mikroprozessoren auf den Markt. Die Mikroprozessoren sind in Prinzip Kleinst - Computer auf wenigen hochintegrierten Siliziumplättchen.

1980 Tauchten die ersten Personal Computer auf.
Seither wuchs die Integrationsdichte, so dass es heute möglich ist, einen ganzen Mikrocomputer auf einem einzigen Siliziumplättchen herzustellen.

1. Programmiersprachen

1.1. Maschinencode: Programmiersprache 1. Generation

Der Maschinencode oder die Maschinensprache ist eine Programmiersprache, die der Computer unmittelbar versteht. Maschinensprachen sind für Menschen schwer verständlich und unübersichtlich. Sie bestehen aus binären oder hexadezimalen Zahlen.

Das könnte etwa so aussehen:

0202 03E8 0203 03E8 0602 1801 10FD

Dieses Programm veranlasst einen bestimmten Computer, auf tausend zu zählen.

Ein Kompromiss sind die Assemblersprachen, in denen die Zahlen durch leicht merkbare Buchstabengruppen dargestellt werden.

Die Maschinensprache ist die unterste Stufe der Computerprogrammierung und die einzige Sprache, welche ein Computer ohne Hilfe eines anderen Programmes ausführen kann.

Programme in Maschinensprache werden für einen bestimmten Rechner oder Rechnertyp entwickelt und sind nicht damit nicht auf anderen Rechnern ausgeführt werden können. D. h. sie sind nicht portabel.

Beispiele von Programmiersprachen:M68000, 386

1.2. Assembler: Programmiersprache 2. Generation

Symbolische, maschinenorientierte Sprache bzw. Übersetzungsprogramm.

Eine Maschinensprache, bei der die einzelnen Befehle durch leicht merkbare Buchstabengruppen dargestellt werden.

Das folgende Assemblerprogramm weist den Computer an, auf tausend zu zählen:

```
        LI    2,1000
LOOP    DEC   2
        JOC   END
        JMP   LOOP
END     END
```

Die Assembler-Sprache stellt für jeden elementaren Maschinenbefehl einen mnemotechnischen, maschinengebundenen Code bereit. Dabei bedeutet „mnemotechnisch" erlernbar, verständlich, indem z. B. für den Maschinenbefehl 03EF der Assemblerbefehl INC benützt wird, was soviel bedeutet wie „addiere 1 zum Inhalt des Rechenregisters".

Das Assemblerprogramm oder der Assembler übersetzt den Source-Code oder Quellkode (Assembler - Code) in Maschinencode um, welcher direkt ausgeführt werden kann.

Auch beim Assembler gilt, dass Assembler - Programme für einen bestimmten Rechner oder Rechnertyp geschrieben werden und damit nicht auf anderen Rechnern ausgeführt werden können.

Diese Schwachstelle versuchte man mit einem Cross-Assembler zu umgehen, indem der Cross-Assembler Assembler-Code in Maschinen-Code eines anderen Rechners

umsetzt. Damit werden Assemblerprogramme (bedingt) portierbar.
Unter „SOURCECODE" wird auch die Datei verstanden, welche die Anweisungen des Programms enthält. Beispiele von Assemblersprachen sind: MAKRO, MASM, Turbo Assembler

1.3. Programmiersprache 3. Generation

Prozedurale Sprachen für spezielle Anwendungsbereiche. Symbolische, problemorientierte Sprachen bei welchen der Ablauf eines Programms durch die Reihenfolge der Anweisungen festgelegt wird (nicht mehr maschinenorientiert, portabel).

1. INTERPRETER
   Ein Maschinenprogramm, das einzelne Befehle einer höheren Sprache in Maschinensprache übersetzen und direkt ausführen kann, also eine Art Simultanübersetzer. Einige höhere Sprachen lassen die Anwendung eines Interpreters nicht zu; das Programm muss als Ganzes übersetzt werden. Dazu dient der Compiler.
   BASIC ist eine Interpreter-Sprache, Pascal nicht. Programme in Pascal muss man mit einem Compiler übersetzen.
   Bei der Programmausführung wird jeweils die aktuelle Programmzeile des Sourcecodes in Maschinen - Code übersetzt und sofort ausgeführt.
2. COMPILER
   Der Sourcecode wird in Maschinencode übersetzt und in Dateien mit Objectcode abgespeichert. Diese Module müssen mit dem Linker zu einem ausführbaren Programm zusammengefügt werden.
3. LINKER
   Der Linker fügt verschiedenen Objectcode mit weiteren Resourcen (Modulen, Bibliotheken, usw.) zu einem ausführbaren Programm zusammen.

Beispiele von Programmiersprachen der 3. Generation sind: BASIC, COBOL, FORTRAN, PASCAL, MODULA-2, ADA, C, OCCAM, PL/I, RPG II usw.

1.4. Programmiersprache 4. Generation

Nichtprozedurale Sprachen für unterschiedliche Anwendungen. Nicht – von – Neumann - Sprachen.

Symbolische, problemorientierte Sprachen, bei welchen Angegeben wird, was das Programm tun soll, ohne jedoch anzugeben, wie dies im einzelnen zu geschehen hat (Deskriptivität, Objekt - Orientiert)

Beispiele von Programmiersprachen der 4. Generation sind: ORACLE, BRICK, SQL (für Informationssysteme), PARADOX, dBase, ObjektVision, C++, TurboPascal V6, MODULA-2 V3, APL.(Integrierte Programme für PC's) FrameWork, Symphony 1-2-3, Open Access.

1.5. Programmiersprache 5. Generation

AI Artificial Inelligence

KI Künstlichen Intelligenz

Beispiele von Programmiersprachen der 5. Generation: LISP, PROLOG, SMALLTALK, LOGO.

1.6. RAD - Rapid Application Developpment

Mit der horrenden Verbreitung graphischer Oberflächen (im Besonderen Windows) wurden vermehrt graphisch und Objekt–orientierte Sprachen bzw. Entwicklungs-umgebungen durch, wie dies z. B. mit Visual Basic von Microsoft auf der einen und Delphi bzw. CBuilder von Inprise (Borland) der Fall ist. Dabei werden alle Windows basierten Teile so gekapselt, dass der Entwickler sich lediglich noch mit seinem Code beschäftigen muss.

1.7. Webbasierte Programmentwicklung

Während in den Anfängen Interpreter kaum eine Chance hatten, erleben diese mit immer schnelleren Prozessoren eine ungeahnte Renaissance, indem Internet-Browser prozedurale Programme langsaam verschwinden lassen. Grundlagen sind die Seitenbeschreibungssprache HTML und Sprachen wie PHP, Java, usw.

Mittlerweile stehen AMP-Systeme kostenfrei zur Verfügung, um lokal einen Webserver (Apache), ein Datenbanksystem (MySQL) und einen PHP Interpreter zu betreiben.

Ebenso stehen verschiedene webbasierte Entwicklungsumgebungen völlig kostenfrei zur Verfügung. Beispiel sind NetBeans, Eclipse, usw.

Gleichzeitig stehen immer mehr Programmpakete oder Frameworks zur Verfügung, welche den Programmieraufwand für Web-Anwendungen auf das absolut wesentliche reduzieren – ein Bespiel ist Laravel.