des Zweiten Weltkriegs nach außerhalb der Stadt verlegt. AT&T war ein kriegswichtiges Unternehmen und bot fachliche Beratung für ein breites Spektrum von militärischen Problemen – selbstverständlich für Kommunikationssysteme, aber auch für die Feuerleitrechner von Flugabwehrgeschützen, für Radar und Kryptografie. Ein Teil dieser Arbeit wurde in Vororten und ländlichen Gegenden von New Jersey erledigt. Der größte Standort befand sich in einem Gebiet namens Murray Hill, das zu den Kleinstädten New Providence und Berkeley Heights gehörte und 33 km westlich des alten Standorts lag.

Abbildung 1.1 zeigt die geografische Lage. 463 West Street befindet sich am Hudson River, auf der Karte ein wenig nördlich der Markierung für den Highway 9A. Die Bell Labs in Murray Hill liegen auf der Grenze zwischen New Providence und Berkeley Heights, etwas nördlich der Interstate 78. Beide Standorte sind mit roten Punkten markiert.

Immer mehr Tätigkeiten der Bell Labs wurden nach Murray Hill verlegt. Der Standort in der West Street wurde 1966 endgültig aufgegeben. In den 60er Jahren arbeiteten in Murray Hill mehr als 3.000 Personen, davon mindestens 1.000 mit einem Doktortitel in einer naturwissenschaftlich-technischen Disziplin wie Physik, Chemie, Mathematik oder verschiedenen Ingenieurswissenschaften.

Abbildung 1.2 zeigt eine Luftaufnahme der Einrichtungen in Murray Hill aus dem Jahr 1961. Aufgeteilt ist die Anlage in drei Hauptgebäude. Nr. 1 befindet sich in der Bildmitte und Nr. 2 oben links. Bei dem Viereck mit dem offenen Innenhof rechts handelt es sich um Nr. 3. Von dem einen Ende von Gebäude 1 bis zum anderen Ende von Gebäude 2 verlief damals ein durchgehender Korridor von 400 m Länge, der erst in den 70er Jahren durch zwei neu errichtete Gebäude unterbrochen wurde.

Ich habe mehr als 30 Jahre in Gebäude Nr. 2 gearbeitet, von meinem Praktikum im Jahr 1967 bis zu meiner Pensionierung 2000. Meine Büros befanden sich in den mit Punkten gekennzeichneten Seitenflügeln im vierten (obersten) Stock. Zur besseren Orientierung bei den folgenden Beschreibungen: Treppenhaus 9 befindet sich am hintersten Ende von Gebäude Nr. 2, Treppenhaus 8 einen Seitenflügel näher im Vordergrund. In den ersten Jahren war der Unix-Raum in der Dachkammer im fünften Stock zwischen diesen beiden Treppenhäusern untergebracht.

Abbildung 1.3 zeigt eine Google-Satellitenaufnahme der Bell Labs aus dem Jahr 2019. Die Gebäude Nr. 6 (unterhalb und links der Bildmitte, markiert) und Nr. 7 (oberhalb und rechts der Bildmitte) wurden Anfang der 70er hinzufügt. Nach 1996 fungierte Gebäude Nr. 6 einige Jahre lang als Hauptsitz von Lucent Technologies. Es ist faszinierend zu sehen, wie viel Firmengeschichte in den von Google hinzugefügten Beschriftungen steckt: Bell Labs als Hauptmarkierung, Lucent Bell Labs auf der Ausfahrt, Alcatel-Lucent Bell Labs an der Einfahrt und Nokia Bell Labs am Anbau der Verwaltungspyramide in Gebäude Nr. 6.

Ich bin nicht wirklich qualifiziert, um eine ausführliche Geschichte der Labs zu schreiben, aber das haben zum Glück schon andere Autoren erledigt. Mir persönlich gefallen besonders das Buch *The Idea Factory* von Jon Gertner, das den Schwerpunkt auf die physikalische Forschung legt, sowie *The Information* von James Gleick, eine hervorragende Quelle für die Tätigkeiten auf dem Gebiet der Informationswissenschaft. Die umfangreiche offizielle Publikation der Bell Labs (fast 5.000 Seiten in sieben Bänden) unter dem Titel *A History of Science and Engineering in the Bell System* ist ausführlich, zuverlässig und für meinen Geschmack äußerst interessant.

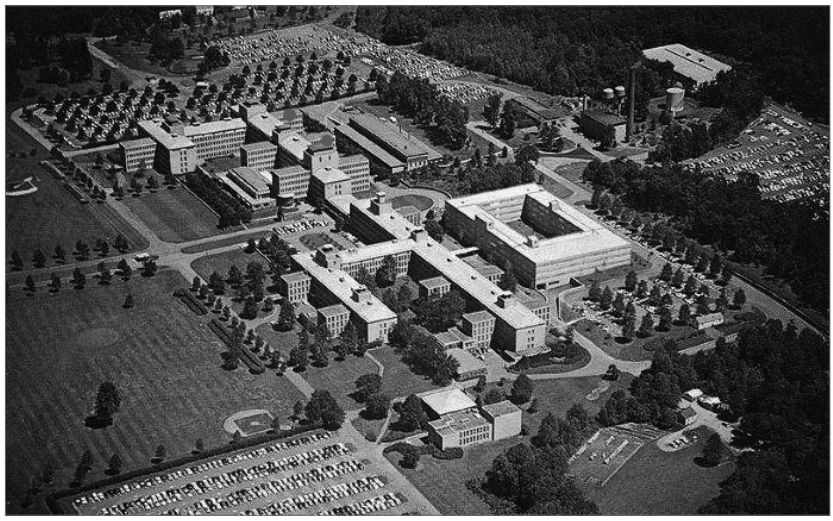

**Abbildung 1.2**
Die Bell Labs im Jahre 1961 (mit freundlicher Genehmigung der Bell Labs)

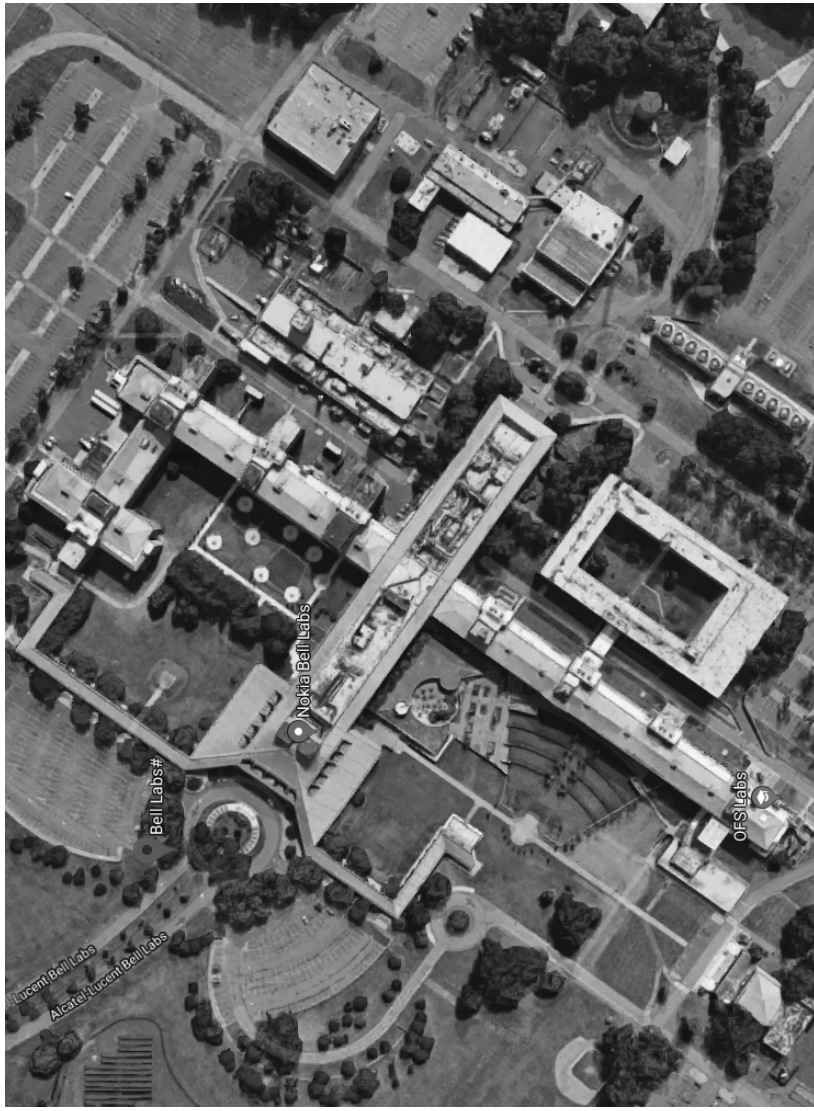


**Abbildung 1.3**
Die Bell Labs im Jahre 2019. Gebäude Nr. 6 liegt unten links.

## Naturwissenschaften in den Bell Labs

In den ersten Jahren drehte sich die Forschung in den Bell Labs vorwiegend um Physik, Chemie, Materialwissenschaft und Kommunikationssysteme. Die Forscher genossen außerordentlich viel Freiheit, um ihren

eigenen Interessen nachzugehen, aber das Umfeld bot so viele relevante Probleme, dass es nicht weiter schwer war, sich auf Gebieten zu betätigen, die sowohl wissenschaftlich interessant waren als auch potenziellen Nutzen für Bell System und für die Welt im Allgemeinen versprachen.

Die Bell Labs steckten hinter einer großen Menge wissenschaftlicher und technischer Fortschritte von umwälzender Bedeutung. An erster Stelle steht dabei der Transistor, der 1947 von John Bardeen, Walter Brattain und William Shockley erfunden wurde, als sie daran arbeiteten, Verstärker für Ferntelefonleitungen zu verbessern. Der Transistor ging aus Grundlagenforschung über Halbleiter hervor, angeregt durch den Bedarf für Geräte, die haltbarer waren als Vakuumröhren und weniger Energie verbrauchten. Letztere bildeten in den 40er Jahren die einzige Möglichkeit, um Kommunikationsgeräte zu bauen – und nebenbei bemerkt auch die ersten Computer.

Die Erfindung des Transistors wurde mit dem Physik-Nobelpreis von 1956 gewürdigt – einem von neun Nobelpreisen für naturwissenschaftliche Arbeiten, die ganz oder teilweise in den Bell Labs erfolgten. Zu den weiteren bedeutenden Erfindungen gehören gegengekoppelte Verstärker, Solarzellen, Laser, Mobiltelefone, Kommunikationssatelliten und CCD-Sensoren (auf deren Grundlage Ihre Handykamera funktioniert).

Von den 60ern bis in die 80er arbeiteten etwa 3.000 Personen im Forschungsbereich der Bell Labs (vor allem in Murray Hill) und 15.000 bis 25.000 in Entwicklungsgruppen an verschiedenen Standorten, an denen Geräte und Systeme für die Bell Labs konstruiert wurden, oft auf der Grundlage von Ergebnissen des Forschungsbereichs. Das ist eine stattliche Anzahl. Wer hat sie alle bezahlt?

Da AT&T Telefondienste für fast die gesamten USA anbot, hatte das Unternehmen praktisch ein Monopol. Allerdings waren seine Möglichkeiten, diese Monopolstellung auszunutzen, beschränkt, da es von Behörden auf Bundes- und Staatenebene reguliert wurde. Die Preise, die AT&T für die verschiedenen Dienstleistungen verlangen durfte, unterlagen einer Kontrolle, und das Unternehmen durfte nur Geschäfte ausüben, die unmittelbar mit der Bereitstellung von Telefondiensten zusammenhingen.

Diese Regulierung funktionierte viele Jahre lang sehr gut. AT&T war verpflichtet, Dienstleistungen für alle anzubieten (»universeller Dienst«),

auch für abgelegene, unrentable Orte. Zum Ausgleich erhielt das Unternehmen eine stabile und vorhersehbare Rendite.

Im Rahmen dieser Vereinbarungen zweigte AT&T einen kleinen Teil seiner Einkünfte an die Bell Labs ausdrücklich für den Zweck ab, die Kommunikationsdienstleistungen zu verbessern. De facto wurden die Bell Labs also durch eine bescheidene Gebühr bezahlt, die bei jedem Telefonat im Land anfiel. Nach einer Abhandlung von A. Michael Noll gab AT&T etwa 2,8 % der Einnahmen für Forschung und Entwicklung aus, wobei etwa 0,3 % auf Grundlagenforschung entfielen. Ich bin mir nicht sicher, ob so etwas auch heute noch funktionieren würde, aber jahrzehntelang sorgte diese Vorgehensweise für kontinuierliche Verbesserungen des Telefonsystems und eine erhebliche Anzahl grundlegender wissenschaftlicher Entdeckungen.

Die sichere Finanzierung war von großer Bedeutung für die Forschung. Dadurch gewann AT&T eine sehr langfristige Sichtweise. Die Forscher bei den Bell Labs hatten die Freiheit, sich auch mit Themen zu beschäftigen, die keinen kurzfristigen Ertrag versprachen und vielleicht sogar niemals gewinnbringend sein mochten. Das unterscheidet sich sehr stark von der heutigen Situation, in der oft nur wenige Monate vorausgeplant und viel Aufwand betrieben wird, um über die finanziellen Ergebnisse im nächsten Quartal zu spekulieren.

## Kommunikationstechnik und Informatik

Die Bell Labs leisteten naturgemäß Pionierarbeit bei der Konstruktion, Erstellung und Verbesserung von Kommunikationssystemen. Dieser Sammelbegriff umfasste alles – von der Endbenutzerhardware wie den Telefonen über Schaltsysteme und Mikrowellen-Sendeanlagen bis zu Glasfaserkabeln.

Manchmal führte diese breite Palette von praktischen Anwendungen sogar zu Fortschritten in der Grundlagenforschung. Beispielsweise versuchten Arno Penzias und Robert Wilson 1964 herauszufinden, was das störende Rauschen verursachte, das bei den von Echo-Ballonsatelliten zurückgeworfenen Funksignalen auftrat. Dabei erkannten sie schließlich, dass es sich um eine Hintergrundstrahlung handelte, die noch vom Urknall herrührte. Diese Entdeckung brachte ihnen 1978 den Nobelpreis ein. Penzias kommentierte dies mit den Worten: »Die meisten Leute

bekommen den Nobelpreis für Dinge, nach denen sie gesucht haben. Wir haben ihn für etwas bekommen, das wir loswerden wollten.«

Eine weitere Aufgabe der Bell Labs bestand darin, ein genaues mathematisches Verständnis der Funktionsweise von Kommunikationssystemen zu entwickeln. Das wichtigste Ergebnis dabei war die von Claude Shannon begründete Informationstheorie, die teilweise auf seinen Kryptografiestudien während des Zweiten Weltkriegs fußte. In seiner 1948 im *Bell System Technical Journal* veröffentlichten Abhandlung »A Mathematical Theory of Communication« beschrieb er die grundlegenden Eigenschaften von Information und fand heraus, wie man die Informationsmenge bestimmt, die prinzipiell über ein Kommunikationssystem übertragen werden kann. Shannon arbeitete von Anfang der 40er Jahre bis 1956 in Murray Hill. Danach kehrte er ans MIT zurück, wo er studiert hatte. 2001 starb er im Alter von 84 Jahren.

Als Computer immer leistungsfähiger und billiger wurden, erweiterten die Bell Labs ihren Einsatz nicht nur im Bereich der Datenanalyse, sondern dehnten ihn auch auf die umfassende Modellierung und Simulation von physikalischen Systemen und Prozessen aus. Die Bell Labs hatten sich schon seit den 30er Jahren mit Computern und EDV befasst und unterhielten seit Ende der 50er Jahre Rechenzentren mit zentralen Großrechnern.

Anfang der 60er wurden Mitarbeiter des Forschungszweiges Mathematik sowie Mitglieder des Bedienungspersonals für den großen Zentralcomputer in Murray Hill einer neu gegründeten Informatik-Forschungsgruppe zugeteilt, die die Bezeichnung Computing Science Research Center erhielt (nach der Nummer in der Organisationshierarchie oft auch »Center 1127« genannt). Auch wenn sie für kurze Zeit noch Computerdienste für alle Mitarbeiter in Murray Hill leistete, war sie keine Serviceeinrichtung, sondern gehörte zum Unternehmensbereich Forschung. 1970 wurde die Gruppe, die sich um die IT-Dienstleistungen kümmerte, in eine eigenständige Organisation ausgelagert.

## Meine Zeit in den Bell Labs

Ein großer Teil dieses Abschnitts erzählt von meiner persönlichen Geschichte. Ich möchte Ihnen damit zeigen, welchen glücklichen Zufällen ich meine Karriere in der Informatik zu verdanken habe und dass die Bell Labs unschlagbar sind, wenn es darum geht, eine solche Karriere zu verfolgen.

Ich wurde in Toronto geboren und ging auch dort zur Universität. Mein Fachgebiet war Technische Physik, praktisch ein Sammelbecken für alle, die nicht genau wussten, auf welchen Schwerpunkt sie sich konzentrieren sollten. Meinen Abschluss machte ich 1964, in der Frühzeit der EDV. Ich war in meinem dritten Studienjahr, als ich zum ersten Mal einen Computer sah, nämlich den Großrechner, der für die gesamte Uni da war. Die IBM 7094 zählte damals zu den Spitzenmodellen und verfügte über einen Magnetkernspeicher für 32K (also 32.768) 36-Bit-Wörter (heute würden wir sagen 128 KByte) sowie Sekundärspeicher in Form großer, mechanischer Plattenlaufwerke. Sie kostete drei Millionen Dollar und war in einem großen Raum mit Klimaanlage untergebracht, behütet von professionellem Bedienpersonal. Gewöhnliche Sterbliche (insbesondere Studenten) wurden nicht einmal in die Nähe gelassen.

Daher habe ich mich während meines Diplomstudiums nur wenig mit Informatik beschäftigt. Allerdings lernte ich die Programmiersprache Fortran. Deshalb kann ich mich nur zu gut in Menschen hineinversetzen, die Schwierigkeiten haben, ihr erstes Programm zu schreiben. Ich hatte zwar Daniel McCrackens hervorragendes Buch über Fortran II gelesen und alle Regeln parat, aber mir war nicht klar, wie ich überhaupt beginnen sollte, um ein Programm zu schreiben. Dieser erste Schritt scheint vielen Einsteigern Schwierigkeiten zu bereiten.

Im Sommer vor meinem letzten Collegejahr hatte ich einen Job bei Imperial Oil in Toronto als Mitglied einer Gruppe, die Optimierungssoftware für Raffinerien entwickelte. (Imperial Oil gehörte zum Teil der Firma Standard Oil of New Jersey, aus der 1972 Exxon wurde.)

Im Nachhinein muss ich zugeben, dass ich wohl ein ziemlich unterdurchschnittlicher Praktikant war. Ich verwendete den ganzen Sommer darauf, ein riesiges Cobol-Programm zur Analyse von Raffineriedaten zu schreiben. An den genauen Zweck kann ich mich nicht mehr erinnern, aber ich weiß noch genau, dass es niemals funktionierte. Mir war nicht klar, wie

man überhaupt programmiert, Cobol bot wenig Unterstützung für eine gute Programmgliederung, und strukturierte Programmierung war noch nicht erfunden. Mein Code bestand aus einer endlosen Folge von `IF`-Anweisungen mit Verzweigungen, bei denen der Code irgendetwas tun sollte, sobald ich herausgefunden hatte, was dieses Etwas war.

Ich versuchte auch, einige Fortran-Programme auf der IBM 7010 von Imperial ans Laufen zu bekommen. Schließlich hatte ich gewisse Kenntnisse in Fortran – auf jeden Fall bessere als in Cobol –, und Fortran hätte sich auch besser für die Datenanalyse geeignet. Erst nachdem ich mich wochenlang mit JCL herumgeschlagen hatte, der »Job Control Language« von IBM, stellte ich fest, dass der 7010 keinen Fortran-Compiler hatte. Die Fehlermeldungen von JCL waren so undurchschaubar, dass es bis dahin auch niemand sonst herausgefunden hatte.

Als ich nach diesem etwas frustrierenden Sommer für mein Abschlussjahr an die Universität zurückkehrte, war ich immer noch sehr stark an Computern interessiert. Es gab keine formalen Informatikkurse, aber dennoch schrieb ich meine Diplomarbeit über künstliche Intelligenz, was damals ein gefragtes Thema war. Sogenannte »Theorembeweiser« oder Programme, die Schach und Dame spielen können, sowie die maschinelle Übersetzung menschlicher Sprachen schienen damals alle in Reichweite zu sein und nur noch ein bisschen Programmierung zu erfordern.

Nachdem ich 1964 meinen Abschluss gemacht hatte, wusste ich nicht recht, was ich als Nächstes tun sollte. Wie viele andere Studenten verschob ich die Entscheidung, indem ich mich zu einem Aufbaustudium entschloss. Ich bewarb mich bei einem Dutzend Universitäten in den USA (was damals unter Kanadiern nicht üblich war – und ich bin aus Kanada) und erhielt von mehreren eine Zusage, darunter vom MIT und von Princeton. Princeton teilte mir mit, dass normalerweise drei Jahre bis zur Promotion vergingen, während es laut Aussage des MIT wahrscheinlich sieben Jahre sein würden. Außerdem bot mir Princeton ein komplettes Stipendium an, während ich beim MIT 30 Stunden die Woche als Forschungsassistent zu arbeiten hätte. Das machte die Entscheidung ziemlich einfach. Außerdem studierte Al Aho, ein guter Freund, der in Toronto ein Jahr über mir gewesen war, bereits in Princeton. Also machte ich mich auf den Weg, was sich später als eine äußerst vorteilhafte Entscheidung herausstellte.

1966 hatte ich erneut Glück, da ich ein Sommerpraktikum beim MIT ergattern konnte. Teilweise lag das daran, dass Lee Varian, ebenfalls Princeton-Student, dort im Vorjahr hervorragende Arbeit geleistet hatte. Den Sommer über nutzte ich das CTSS (Compatible Time-Sharing System) und schrieb Programme in MAD (Michigan Algorithm Decoder, ein Dialekt von Algol 58), um Tools für das neue Betriebssystem Multics zu schreiben, auf das ich in Kapitel 2 noch zurückkommen werde. (Multics wurde ursprünglich MULTICS geschrieben, aber die Variante mit Kleinbuchstaben wirkt optisch angenehmer. Ich verwende hier die besser lesbare Version, auch wenn sie historisch nicht korrekt ist. Das gilt auch für andere komplett in Großbuchstaben geschriebene Wörter, also z. B. UNIX/Unix.)

Mein offizieller Vorgesetzter beim MIT war Prof. Fernando Corbató, von allen nur »Corby« genannt, ein wunderbarer Gentleman, der nicht nur CTSS entworfen hatte, sondern auch für Multics verantwortlich war. Für seine grundlegenden Arbeiten an Timesharing-Systemen erhielt er den Turing Award 1990. Corby starb im Juli 2019 im Alter von 93 Jahren.

Corby leitete nicht nur das Design und die Implementierung von CTSS und Multics, sondern war auch der Erfinder von Passwörtern für den Computerzugriff. Im Stapelverarbeitungsbetrieb war so etwas nicht nötig gewesen, aber bei einem Mehrbenutzercomputer mit einem gemeinsamen Dateisystem war ein Sicherheitsmechanismus zum Schutz der privaten Dateien erforderlich.

Ich kann mich noch an die beiden kurzen, aussprechbaren Passwörter erinnern, die für mich im Sommer 1966 am MIT generiert wurden. Es waren zwei, nicht nur eines, da eines Tages jemand die Datei mit den Passwörtern und die Datei mit der Login-Meldung des Tages verwechselte. Da die Passwörter unverschlüsselt gespeichert waren, konnte dadurch jeder, der sich anmeldete, sämtliche Passwörter einsehen.

Mein Büro gehörte eigentlich Prof. Joseph Weizenbaum, der den Sommer über aber nicht da war, sodass ich es nicht mit ihm teilen musste. Anfang 1966 hatte Weizenbaum eine Beschreibung von Eliza veröffentlicht, einem Programm, das ein Gespräch mit einem Psychologen der Rogers-Schule simulierte. Eliza stellte einen der ersten Versuche dar, zur Interaktion mit einem Computer auf natürliche Sprache zurückzugreifen. Die Simulation wirkte erstaunlich überzeugend, zumindest wenn man

ihr zum ersten Mal ausgesetzt wurde, und ich hatte großen Spaß dabei, sie Besuchern stolz vorzuführen. Heute sind im Web viele Eliza-Simulatoren zu finden. Es ist immer noch recht vergnüglich, ein paar Minuten damit herumzuspielen, allerdings werden ihre Schwächen sehr bald deutlich.

```
ELIZA: Please tell me what's been bothering you.
SIE:   I'm trying to write a book about Unix.
ELIZA: Is it because you are trying to write a book
       about unix that you came to me?
SIE:   Yes.
ELIZA: You are sure.
SIE:   Yes.
ELIZA: You seem to be quite positive.
SIE:   Yes.
ELIZA: I see.
```

Im Sommer 1967 zog ich dann das große Los: Ich bekam eine Praktikumsstelle bei den Bell Labs in Murray Hill, wo ich im Computing Science Research Center für Doug McIlroy (Abbildung 1.4) arbeitete. Doug schlug mir vor, ein Problem bei der Beurteilung von Speicherzuweisungsverfahren zu untersuchen, was eines seiner langfristigen Forschungsgebiete war. In typischer Praktikantenmanier wurstelte ich also so herum und machte letzten Endes etwas völlig anderes – nämlich eine Bibliothek von Funktionen zur Vereinfachung der Listenverarbeitung in Fortran-Programmen aufzustellen. Den Sommer über schrieb ich Programme in einer strengen Assemblersprache für den damals aktuellen Großrechner in Murray Hill, einen GE 635. Dabei handelte es sich im Grunde genommen um eine verbesserte IBM 7094, aber auch um eine vereinfachte Version der GE 645, die eigens für Multics entworfen worden war. Das war so ziemlich die letzte Gelegenheit, bei der ich in Assemblersprache schrieb. Die ganze Sache war zwar völlig verfehlt, machte aber einen Riesenspaß, und damit war ich der Programmierung ein für alle Mal verfallen.

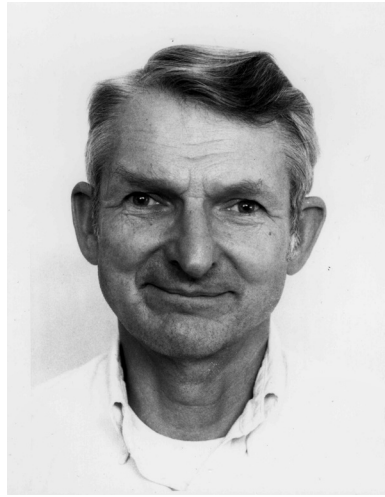

**Abbildung 1.4**
Doug McIlroy, ca. 1984 (freundlicherweise zur Verfügung gestellt von Gerard Holzmann)

## Die Büros

Die räumliche Anordnung kann manchmal schicksalhaft sein.

Das Büro, das ich 1967 als Praktikant bezog, lag im vierten Stock von Gebäude Nr. 2 an einem Flur, der von Treppenhaus 8 abzweigte. An meinem ersten Tag saß ich in diesem Büro (das waren noch die guten alten Zeiten, in denen man selbst als Praktikant mit Glück ein Büro für sich allein ergattern konnte) und fragte mich, was ich hier eigentlich tun sollte, als ein älterer Herr um 11 Uhr seinen Kopf zur Tür hereinstreckte und sagte: »Hallo, ich bin Dick [unverständlich]. Sollen wir Mittagessen gehen?«

Also gut, dachte ich, warum nicht? Von der Mittagspause selbst weiß ich nichts mehr, aber ich kann mich noch gut daran erinnern, dass Dick [unverständlich] nachher noch anderswo hinging und ich den Flur entlangschlich, um das Namensschild an seiner Bürotür zu lesen. Richard Hamming! Mein freundlicher Zimmernachbar war der berühmte Erfinder von Fehlerkorrekturcodes und der Autor des Lehrbuchs für einen Kurs in numerischer Analyse, den ich gerade erst belegt hatte.

Dick (Abbildung 1.5) und ich wurden gute Freunde. Er hatte feste Überzeugungen und scheute sich auch nicht, sie auszudrücken, auch wenn er

damit einige Leute vor den Kopf stieß. Ich dagegen genoss seine Gesellschaft, und sein Rat war für mich äußerst nutzbringend.

Zwar war er Abteilungsleiter, allerdings gab es in seiner Abteilung keinen einzigen Mitarbeiter, was ein bisschen merkwürdig erschien. Er erzählte mir, dass er hart gearbeitet hatte, um diese Kombination aus angemessenem Titel und fehlender Verantwortung zu erreichen. Nachvollziehen konnte ich das erst, als ich selbst Leiter einer Abteilung mit einem Dutzend Mitarbeitern wurde.

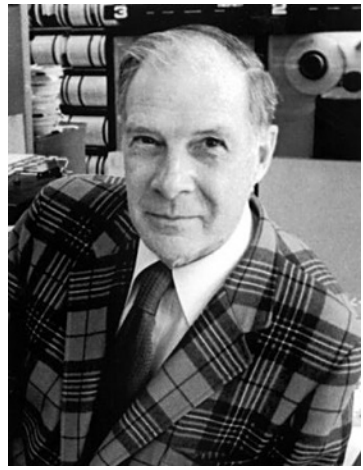

**Abbildung 1.5**
Dick Hamming, ca. 1975, mit seinem typischen Plaidjacket (Wikipedia)

Ich war dabei, als er im Sommer 1968 erfuhr, dass er den ACM Turing Award gewonnen hatte, der heute als das Gegenstück zum Nobelpreis in der Informatik angesehen wird. Damals war der Nobelpreis mit 100.000 Dollar dotiert, der Turing Award dagegen mit nur 2.000 Dollar, was Dick zu der sarkastischen Bemerkung veranlasste, er hätte 2% des Nobelpreises gewonnen. (Dies war übrigens erst die dritte Verleihung des Turing Award. Die beiden vorhergehenden Preise gingen an Alan Perlis und Maurice Wilkes, ebenfalls Pioniere der Informatik.) Dick wurde für seine Arbeiten über numerische Methoden, automatische Codiersysteme sowie Codes zur Fehlererkennung und -behebung ausgezeichnet.

Es war auch Dick, der mich dazu anregte, Bücher zu schreiben, was sich als eine gute Sache herausstellte. Er hielt nicht viel von den meisten Programmierern, da sie seiner Meinung nach wenn überhaupt, dann nur

eine schlechte Ausbildung erhalten hatten. Ich kann ihn immer noch sagen hören:

> *»Wir geben ihnen ein Wörterbuch und die Grammatikregeln an die Hand und sagen ihnen: ›Junge, jetzt bist du ein großartiger Programmier.‹«*

Seiner Ansicht nach sollte Programmieren ebenso gelehrt werden wie Schreiben. Was guter und was schlechter Code war, sollte auch anhand des Stils unterschieden werden. Programmierer sollten lernen, gut zu schreiben und guten Stil zu würdigen.

Unsere Meinungen gingen zwar darüber auseinander, wie das zu erreichen war, aber sein Grundgedanke war sinnvoll und führte unmittelbar zu meinem ersten Buch, *The Elements of Programming Style*, das ich 1974 zusammen mit P. J. »Bill« Plauger veröffentlichte, der zu diesem Zeitpunkt im Büro nebenan saß. Dabei folgten Bill und ich dem Vorbild des englischen Stilhandbuchs *The Elements of Style* von Strunk und White, indem wir ebenfalls Beispiele von schlecht geschriebenem Code vorstellten und zeigten, wie sie sich jeweils verbessern ließen.

Unser erstes Beispiel stammte aus einem Buch über numerische Analyse, das Dick mir gezeigt hatte. Eines Tages kam er damit in mein Büro und ließ sich völlig aufgebracht darüber aus, wie schlecht die numerischen Programmteile waren. Mir sprang gleich ein Abschnitt mit furchtbarem Fortran-Code ins Auge:

```
      DO 14 I=1,N
      DO 14 J=1,N
14 V(I,J)=(I/J)*(J/I)
```

Lassen Sie mich das für Nicht-Fortran-Programmierer erklären. Dieser Code besteht aus zwei verschachtelten `DO`-Schleifen, die beide in der Zeile mit der Nummer 14 enden. Jede der Schleifen durchläuft ihre Indexvariable von der unteren zur oberen Schranke. In der äußeren Schleife durchläuft also `I` die Werte von 1 bis `N`, in der inneren Schleife `J`. Die Variable `V` ist ein Array aus `N` Zeilen und `N` Spalten. `I` durchläuft alle Zeilen und `J` jeweils alle Spalten einer Zeile.

Dieses Schleifenpaar erstellt also eine `NxN`-Matrix mit dem Wert 1 in der Diagonalen und 0 an allen anderen Positionen. Für `N` = 5 ergibt sich beispielsweise: