Greetings to the Participants at "Strachey 100" by Dana S. Scott

I am very sorry indeed not to be able to make the trip from Berkeley, California, to be at this birthday party. And it is certainly hard to imagine that Christopher would be 100 years old today! It is even harder to think that he died over 40 years ago at the age of 58 in May of 1975. I can still remember the deep, deep pain I felt in getting the not unexpected, but terrible news from the hospital, and the pain felt by all his circle of relatives, friends, students, and colleagues.

On my recent lecture tour in the UK last summer, I remarked that ***if*** one lives long enough, ***then*** one becomes something of "an historical figure"! And that is rather how I feel today speaking to you at age 84. People say, "*Oh, you knew Christopher Strachey!*" Or ask, "*You really met Kurt Gödel?*" Or, "*You really once visited Bertrand Russell?*" Or, "*What was Paul Erdős like?*" And though I never had a chance to meet Alan Turing (1912—1954), I met many, many people who did, including my colleague and good friend Robin Gandy (1919—1995). He was Turing's only Ph.D. student, remember, and, sadly, he also is now gone for over 20 years.

Let me give you all a ***warning*** and, perhaps, some ***advice***. Having met so many remarkable people in my career, I have that very unpleasant reaction — in reading their obituaries, for example — of asking myself: "*Why in heaven's name didn't you ask them more questions about their lives?*" We are often all too content with quite trivial, social conversations, but a pointed question will sometimes elicit fascinating stories. Now, a remarkable Oxford figure, Sir Isaiah Berlin (1909— 1997), who was a favorite guest of ours when we lived in Oxford, never needed any prodding and was always full of good stories. Sometimes maybe Sir Isaiah indulged too much in what I like to call ***creative remembering***, but his entertainment value was always high. The lives of others, however, are often hidden, and they are the shy ones needing our encouragement to reminisce. So my advice is this: *Don't miss the chance of asking questions! It is much too easy to be too late.*

Fortunately Berlin has an extensive biography by Michael Ignatieff, and Strachey has a quite detailed, briefer biography by Martin Campbell-Kelly. A very detailed history of the Strachey family was written by Barbara Caine; while Christopher's sister, Barbara Strachey, has well documented the "remarkable" female side of the Stracheys and their relations. I mention some of these references in my bibliography.

What, we can wonder, would Christopher have been like at age 100? What indeed. Well, assuming he would have retained all his faculties, he

definitely would have been ***outspoken***, ***critical***, and ***funny*** — in other words he would have been quite the same as he was in his late 50s! But, perhaps, today as an ancient, respected guru, he would have allowed himself to be rather more ***forgiving***. We can only conjecture.

Of course, the whole aspect and style of the computing profession has completely changed over these last 40 years. We need to remember that Christopher was ***not*** in favor of an undergraduate degree. He was ***not*** in favor of the term "Computer Science". And he ***insisted*** on being called "Professor of Computing". And so it is somewhat difficult to think of how his attitudes would have changed now that we have something called ***machine learning*** winning Go games and making money in the stock market (to name only two recent areas of development that scare us).

And would he have liked ***the cloud*** and the extensive remote use of software? (I hope so.) Would he have approved of students getting an undergraduate degree and then opting for the large salaries rather than continuing to a Masters or even a D.Phil.? (I hope not, and I don't.) But too much is out of the hands of us academics today, and we have to learn to adjust. I think Christopher would have adjusted eventually, but he would have made us ***debate*** the changes — because he liked to force people to think.

Let us note that in this year, 2016, the CWI, Amsterdam, will also celebrate the 100th anniversary of Adriaan van Wijngaarden (1916-1987). He was the irrepressible force behind the design of Algol 68. One of van Wijngaarden's early students was Edsger W. Dijkstra (1930—2002), who received his Ph.D. in 1959. I knew Dijkstra and always found his highly critical and judgmental attitude hard to deal with. Aad van Wijngaarden was not al all like that, though he had great self-confidence. One of his later students, Jaco de Bakker (1939-2012), who received his Ph.D. in 1967, I liked enormously. Jaco had a long and distinguished career as a researcher and teacher, both at the CWI (Centrum Wiskunde & Informatica) in Amsterdam and at the Vrije Universiteit.

I met van Wijngaarden and de Bakker in Amsterdam during the academic year 1968/69 while on sabbatical from Stanford University. Here is my favorite story about van Wijngaarden. The design of Algol 68 had been concluded and the report on the language came out the next year. When I received it back in the States in 1970 I was amazed by its complex typesetting, for it had been completely typed on an IBM ® Selectric typewriter using at least a dozen "golfballs" to get all those special symbols and typefaces. On my next visit to Amsterdam when I met van Wijngaarden I said to him: "*Aad, Aad! How in the world did you get a secretary to type such a complicated document?*" He smiled and answered: "*Oh, I have a secretary who types whatever I want!*" And he

silently pointed to himself! Yes, he alone typed it all so he would know it was correct. That takes a truly remarkable determination.

Back in the early 1960s at Berkeley I had been introduced to Algol 60 and was very intrigued by its high-level design, having as a graduate student at Berkeley and Princeton done some (quite small-scale) machine-language programming projects. Then in the mid 1960s at Stanford, when the Computer Science Department was just being formed, I had many connections with the new group of students there and with John McCarthy (1927-2011). McCarthy had received his Ph.D. in 1951 at Princeton, not in Logic but in ***Partial Differential Equations*** under the direction of Solomon Lefschetz (1884-1972). He was however influenced by what he learned of the λ-calculus of Alonzo Church (1903-1995), another Princeton professor at the time.

McCarthy later introduced ***functional abstraction*** into his design of LISP, but on his own admission he did not understand much of Church's logical development of the λ-calculus. I, on the other hand, had as an undergraduate about 1952 at Berkeley learned about the subject on my own from a small, but rich logic text by Paul C. Rosenbloom (1920-2005), and I subsequently studied the monograph of Church (1941), the works of his close colleague Haskell Brooks Curry (1900-1982), and of his prominent students, Stephen Kleene (1909-1994) and J. Barkley Rosser (1907-1989), all of whom I got to know well in later life. However, during my time at Berkeley and Stanford I did not see the role λ-calculus would grow to play in programming-language theory.

During the spring of 1969 de Bakker and I worked on what we called "***A Theory of Programs***" defined in the spirit of Automata Theory, but with ***input*** and ***output***. I reported on the ideas when visiting the Vienna IBM Laboratory later that summer. The notes, handwritten by me, were finally reproduced in 1989 in a Festschrift for Jaco. That visit to Vienna was the start of many long-lasting friendships with the group there, who are now scattered around Europe.

But, more fatefully, I attended my first meeting of the IFIP Working Group WG 2.2 ***Formal Description of Programming Concepts.*** I had been proposed for membership by my long-standing friend and mentor, Patrick Suppes (1922-2014), Professor of Philosophy at Stanford. He was unable to attend that meeting and sent me in his place. Alas, I have lost track of all my files, but some of the history of WG 2.2 is available on the internet. Here is the outline of the very early days:

• IFIP Working Conference on Formal Language Description Languages, met in Vienna, 15-18 September 1964, organized by Prof. Hans Zemanek (1920-2014), leader of the IBM Laboratory. This was the seminal event for the creation of WG 2.2.

• The first meeting of WG 2.2. was held at Alghero, Italy, 6-8 September 1967 organized by Prof. A. Caracciolo di Forino. The first chairman, until the Geneva meeting, was T.B. Steel Jr.

• The next meetings were at Copenhagen, Denmark, 22-26 July 1968, and then at Vienna, Austria, 21-25 April 1969, organized by Prof. Kurt Walk of the IBM Laboratory.

• The meeting I attended was at Colchester, England, 8-12 September 1969, organized by Prof. John Laski.

The 1969 meeting was where I first met Christopher, and I was immediately impressed by the clarity of his statements. These working group conferences always had a lot of arguing back and forth (though not as bad as what I heard about the Algol meetings in WG 2.1!), and Christopher was a very good debater. His experience with the sadly failed CPL (Cambridge Programming Language) project, and his subsequent work, especially in collaboration with Peter Landin (1930-2009), had made him very aware of good design decisions. The Algol language had originally been only meant as a ***publication*** language for numerical algorithms and was not originally intended to become a programming language. However, it's clean ideas attracted many groups to make it — or something like it — an operational language. I note in passing there was a very well attended 40th Anniversary Meeting of WG 2.2 held in Udine, Italy, in September of 2006.

Let us now return to λ-calculus and Strachey's use of it. Christopher told me once that Roger Penrose (now Sir Roger!) suggested to him that he ought to look into using the λ-calculus for the kind of function definitions he wanted to do. At this moment I cannot track down or verify the story. (Perhaps people in Oxford might ask Penrose personally about this?) But, in Landin's three papers in 1964 and 1965, Peter from the start brings in λ-calculus — but without attribution to Strachey, even though they were close associates. It is of course quite possible that Landin thought of this himself, and he mentions Rosenbloom in his references. Landin in one abstract says:

> This paper is a contribution to the "theory" of the activity of using computers. It shows how some forms of expression used in current programming languages can be modelled in Church's λ-notation, and then describes a way of "interpreting" such expressions. This suggests a method of analyzing the things computer users write, that applies to many different problem orientations and to different phases of the activity of using a computer. Also a technique is introduced by which the various composite information structures involved can be formally characterized in their essentials, without commitment to specific written or other representations.

And at the end of his third paper he writes:

> This paper was written while I worked for Christopher Strachey. It is based on some lectures that George Coulouris invited me to contribute in Spring 1963 to a series on the "Logical Foundations of Programming" at the University of London Computer Unit. W. H. Burge, C. Strachey and M. Woodger read parts of an earlier version and pointed out many errors
> and obscurities.

So, there is no question that many ideas were shared. In any case, when I met Strachey and subsequently arranged to spend the Autumn term in Oxford in 1969 working with him, he was using λ-calculus in the style of Landin's papers, which were much read by all of us wanting to follow the paths Landin had convincingly set forth.

Form the beginning of our association in Oxford I told Christopher, "***Lambda-calculus has no mathematical models!***" Calling on my background in model theory and recursion theory, I suggested a partial-order structure for functions and functionals that had been much employed could be adapted to his objectives. There were several earlier works doing computability theory in this way, though at that time the theory people were much more involved in understanding ***infinitary*** computation. Using inspiration from the Ph.D. thesis of Richard Platek and writings of Kleene, I suggested a formulation for Strachey to use in my draft paper "*A type-theoretical alternative to ISWIM, CUCH, OWHY.*"

Then, one Saturday morning in November of 1969, when lying on the bed in the guest room of the flat that I and my family had rented in Headington, an awful thought occurred to me. In setting up the higher types of continuous functionals, I had seen that each of the type domains had a ***basis of finite*** elements — in the sense that each object of each type is the least upper bound of the finite elements it contains. In passing from one type level to the level of continuous functions above it, the basis of finite elements usually became ***more complicated***. But what if, I suddenly thought, there were a type D such that the basis of the function space (D → D) was ***no more*** complicated than the basis of D? Could we then have a type D = (D → D) in the sense of an isomorphism of partially ordered sets? In a fever I set about finding such a domain over the next few days, and that was how Domain Theory was born in my mind. After being so critical of the formal approach to λ-calculus, I would now have to eat my own words, because the λ-calculus could indeed have ***mathematical*** models.

In many ways I rather regret having cast the foundations of Domain Theory at that time in the form of ***lattice theory*** (and later ***DCPO theory***). Some people liked it, and some people disparaged it as "***Scottery***" and too abstract. A change came in the mid 1970s when I had become a colleague of Christopher's at Oxford as the first Professor of Mathematical Logic. Gordon Plotkin, then a new Ph.D. in AI at Edinburgh, sent me a 1972 memo, "*Set-theoretical and other elementary models of the* λ*-calculus,*" inspired by Domain Theory. At that time I at first did not fully understand Plotkin's simplification (by focussing on one particular domain), and I put the memorandum aside. But later for teaching purposes I went back to it and realized that, by using ideas of Recursive Function Theory, the set theory could be replaced just by the ***powerset*** of the set of natural numbers. I wrote up a report on the approach in 1975, which was published the next year, called, "Data types as lattices: To the Memory of Christopher Strachey, 1916-1975." That paper had considerable historical detail and suggestions about understanding higher types. Of course, by then Christopher was gone. My earlier unpublished memo on the type-theoretical alternative, and Plotkin's memo on the set-theoretical model were finally published in the Festschrift for our friend and colleague Corrado Böhm in 1993 — along with additional materials.

And here is another deep regret of mine. I was a graduate student at Princeton in Mathematics from 1955 to 1958 working under the direction of Church. Now he never spoke to us students at that time about λ-calculus. My conjecture is that he was always deeply disappointed that his early idea about a new foundation for logic and mathematics proved to be inconsistent, as his students Kleene and Rosser discovered. He wrote up the ***equational part*** of the theory (proved consistent by syntactical arguments) in 1941 and then turned to ***typed*** λ-calculus, and this later work became very influential indeed. Curry, however, continued to do research in the ***untyped*** theory all his life, while Rosser and Kleene turned away from it after their graduate school period.

What then is my regret? Well, in 1955 John R. Myhill (1923-1987) and John Shepherdson (1926-2015) published their paper "*Effective operations on partial recursive functions".* And in 1956 Richard M. Friedberg and Hartley Rogers Jr. (1926-2015) wrote an abstract on what became their 1959 paper, "*Reducibility and completeness for sets of integers".* Friedberg and Rogers called their operators ***enumeration operators***, but they were basically the same as Myhill and Shepherdson's functionals. And it took me almost 20 years ***until 1975*** to see that they had therefore ***already discovered*** a model for the λ-calculus — but none of them ever seemed to know it!

Since I knew Myhill and Rogers personally and had read Shepherdson's other work, and since I had a good background in recursive function theory and model theory from my student days, and since Kleene was resident in Princeton

for 1956/57, ***and*** since Rodgers visited Princeton that academic year to lecture, ***I myself could have put two and two together to produce a model in 1957!*** If only I had done so and then announced it at the big 1957 Summer Logic Conference at Cornell University, I would now be ***rich*** and ***famous*** (though the famous part is not my principal regret). Also the development of the theory of programming languages would have been much different. Alas, history is not an experimental science, and we cannot turn the clock back to correct our oversights and erase our stupidities.

Another regret is that my taking up the Oxford chair did not start up further collaboration with Strachey. There were two reasons. For better or worse Oxford is governed by committees of academics. Strachey had been given a personal chair in the early 1970s, and my chair was in both Philosophy (under the old Lit Hum Board) and in Mathematics (with its own board). That meant both Christopher and I had to take part in many committee meetings. Also the eight-week Oxford terms went by at a lightening pace. There were also many students around (fortunately!) requiring supervision, but one term owing to leaves of absence I had to supervise 19 postgraduates.

But a second reason that limited collaboration was the decision Strachey made to submit a book to Cambridge for the Adams Prize. The rule was that all authors had to be former Cambridge students or fellows, and so, as I could not be a co-author, Christopher recruited Robert Milne. The resulting Strachey-Milne opus was perhaps too dense and too technical, and in the event it failed to garner the prize — a sad, disheartening fact Christopher learned in hospital just before he died.

Fortunately, Strachey's loyal assistant, Joseph E. Stoy, stepped up to the task and wrote "Denotational Semantics: The Scott-Strachey Approach to Programming Language Theory." That was an excellent presentation, but it would have perhaps been even better with three authors.

At the time of my discovery of the λ-calculus model in 1969, John Reynolds (1935-2013) was visiting England and attended my first lectures about the approach. He became an enthusiastic advocate, as he had been working himself on programming-language design. Over the years he subsequently made many major contributions to design and theory, as recounted in the Brookes-O'Hearn-Reddy Reynolds memorial paper in 2014.

Reynolds, in works with his Ph.D students and, later, with Peter W. O'Hearn, brought a fresh new approach to Denotational Semantics. In the memorial paper the authors write:

> These type constructors are parametrized by a type variable S representing the set of states for a store. This is in marked contrast

> to Strachey's denotational semantics where the store is fixed and global. Here the store is a type variable. In essence, this means that the program can work with whatever "type" of store we supply, as long as it has all the required data variables. An intuitive interpretation of parametric polymorphism immediately implies that the program will act the same way for all these different "types" of stores.

Ah, if only that could have been discovered in the 1970s, the influence of Denotational Semantics would be much greater, as the original Strachey approach to modeling the store was quite *ad hoc* and often unconvincing. Fortunately, however, research continues vigorously today. Most recently O'Hearn has been co-recipient with Stephen Brookes of the 2016 Gödel Prize for the invention of Concurrent Separation Logic, which was deeply influenced by Reynold's ideas and results.

When I was invited to speak at the Princeton University and the Association for Computing Machinery's Alan Turing Centennial Celebrations in 2012 I tried to make a ***time line*** of events and ideas in Logic, Computability, and Category Theory. This is only a faint approximation to a history of ideas, but that might encourage others to flesh out my outline. I also just found through an internet search the slides of a talk by Pierre-Louis Curien on the semantics of programming languages. He and the French have made major contributions to the area, and his report might also be an inspiration to historians.

In conclusion, then, we can agree this has been an amazing half century in the development of computers and the ways of using them. New surprises pop up every week. Ten years ago we could not have predicted even half of what is available today. And, of course, I am deeply sad that Christopher could not live to see what has happened.

Thanks, and my best greetings this November of 2016.

A VERY SELECT BIBLIOGRAPHY

1941
Alonzo Church, "The Calculi of Lambda Conversion." Princeton University Press, 1941, 77 pp.

1950
Paul C. Rosenbloom, "The Elements of Mathematical Logic." Dover, 1950, iv + 214 pp.

1964
Peter J. Landin, "The mechanical evaluation of expressions". The Computer Journal, vol. 6 (1964), pp. 308—320.

1965
Peter J. Landin, "Correspondence between ALGOL 60 and Church's Lambda-notation: part I". Communications of the ACM, vol. 8 (1965), pp. 89—101.

Peter J. Landin, "A correspondence between ALGOL 60 and Church's Lambda-notation: part II". Communications of the ACM, vol. 8 (1965), pp. 158—165.

1966
Peter J. Landin, "The next 700 programming languages". Communications of the ACM, vol.9 (1966), pp. 157—166.

1969
J.W. de Bakker and Dana Scott, “A theory of programs: An outline of joint work.” Handwritten notes presented at an IBM Seminar, Vienna, August 1969. Reprinted in: J.W. Klop, J.J.C. Meijer, and J.J.M.M. Rutten (eds.), “J.W. de Bakker, 25 Jaar Semantiek: Liber Amicorum.” CWI Amsterdam, 437 pp.

Dana S. Scott, “A type-theoretical alternative to ISWIM, CUCH, OWHY.” Informally distributed, October 1969. Published with additions by the author in: Theoretical Computer Science, vol. 121 (1993), pp. 411—420.

1972
Gordon Plotkin, "Set-theoretical and other elementary models of the λ-calculus." Theoretical Computer Science, vol. 121 (1993), pp.351-409, including the previously unpublished 1972 memorandum.

1976
Dana Scott, “Data types as lattices: To the Memory of Christopher Strachey, 1916-1975.” SIAM Journal of Computing, vol. 5 (1976), pp. 522-587.

1977
Joseph E. Stoy, “Denotational Semantics: The Scott-Strachey Approach to Programming Language Theory.” MIT Press 1977, xxx + 414 pp.

1980
Barbara Strachey, “Remarkable Relations: The Story of the Pearsall Smith Family.” London, Victor Gollancz, 1980, 351 pp.

1985
Martin Campbell-Kelly, "Christopher Strachey, 1916-1975: A Biographical Note." IEEE Annals of the History of Computing, vol. 1 (1985), pp. 19-42.

1990
C.A. Gunter and D.S. Scott, “Semantic Domains.” In: “Handbook of Theoretical Computer Science, Volume B,” edited by J. van Leeuwen, North Holland, 1990, pp. 633-674.

1998
Michael Ignatieff, “Isaiah Berlin: a life.” Metropolitan Books, 1998, 356 pp.

2000
Dana Scott, “Some Reflections on Strachey and His Work.” Higher-Order and Symbolic Computation, vol. 13 (2000), pp. 103—114.

2005
Barbara Caine, “Bombay to Bloomsbury: A biography of the Strachey family.” Oxford University Press, 2005, xvii + 488 pp.

2009
Richard Bornat , “Peter Landin: a computer scientist who inspired a generation, 5th June 1930 - 3rd June 2009.” Formal Aspects of Computing, vol. 21 (2009), pp. 393—395.

2011
Pierre-Louis Curien, “A journey into the semantics of programming languages.” Slides for a talk on 24 May 2011 at the Institute of Software of the CAS, Beijing, China, 49 pp.

2012
Dana S. Scott, “A Timeline for Logic, λ-Calculus, and Programming Language Theory.” Slides for talks prepared for: TURING CENTENNIAL CELEBRATION, Princeton University, May 10-12, 2012, and ACM TURING CENTENARY CELEBRATION, San Francisco, June 15-16, 2012, 9 pp.

2014
Dana S. Scott, “Stochastic λ-calculi: An extended abstract.” Journal of Applied Logic, vol. 12 (2014), pp. 369—376.

Dana S. Scott, “Types and Type-Free λ-Calculus.” Slides for various lectures, 28 pp.

Stephen Brookes, Peter W. O’Hearn, and Uday Reddy. "The essence of Reynolds." Formal Aspects of Computing, vol. 26 (2014), pp. 435-439.

Postscript: This talk was not meant as a complete history. There are easily 150 other names that ought to be added to a description of the development over half a century of programming-language definition and semantics — and offshoots from that research. The present author is incapable of writing such a history and only wanted to trace some of the paths he was personally involved in that were directly inspired by Christopher Strachey.