Origins of Recursive Function Theory

STEPHEN C. KLEENE

For over two millenia mathematicians have used particular examples of algorithms for determining the values of functions. The notion of "λ-definability" was the first of what are now accepted as equivalent exact mathematical descriptions of the class of the functions for which algorithms exist. This article explains the notion and traces the investigation in 1931–1933 by which the notion was quite unexpectedly so accepted. The Herbrand-Gödel notion of "general recursiveness" in 1934 and the Turing notion of "computability" in 1936 were the second and third equivalent notions. Techniques developed in the study of λ-definability were applied in the analysis of general recursiveness and Turing compatability.

Keywords: *λ-definability, undecidable sentences, general recursiveness, partial recursiveness, Turing computability, finite automata, recursion theorem, primitive recursiveness, algorithms, Church's thesis, degrees of unsolvability, normal form theorem, arithmetical hierarchy, analytic hierarchy, hyperarithmetical hierarchy, relative recursiveness, inductive definitions*

CR Categories: *1.2, 5.22, 5.26, 5.27*

I could entitle this paper "Four Dozen Years in Recursionland" (1979 − 1931 = 48). When I was invited to lecture at the FOCS symposium, it was indicated that my hearers would be interested in how recursion theory (and the theory of regular events in finite automata theory) originated, as viewed through the eyes of one who was there.

I began my intensive study of the foundations of mathematics with the course given by Alonzo Church at Princeton in the fall semester of 1931–32. My only previous acquaintance with the area had been very general, from Alfred North Whitehead's *An Introduction to Mathematics (1911)*[1] and Bertrand Russell's *Introduction to Mathematical Philosophy (1919)*.

Church's course consisted, except for one interpolation, in the presentation of the contents of his two papers "A set of postulates for the foundation of logic" *1932* and *1933*.

© 1981 by the American Federation of Information Processing Societies, Inc. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the AFIPS copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the American Federation of Information Processing Societies, Inc. To copy otherwise, or to republish, requires specific permission.

Author's Address: Department of Mathematics, Van Vleck Hall, University of Wisconsin, Madison, WI 53706.

Note: This is a revised version of a paper presented at the twentieth annual IEEE Symposium on Foundations of Computer Science, October 1979, San Juan, Puerto Rico.

© 1981 AFIPS 0164-1239/81/01052-67$01.00/0

The interpolation? During the fall of 1931, John von Neumann was the speaker one day at the mathematics colloquium. He chose to speak, not on work of his own, but on Gödel's results on formally undecidable sentences; von Neumann had received a preview of them at a meeting at Königsberg in September 1930, which von Neumann and Gödel both attended (Gödel *1931-32a*). This topic was thereupon incorporated into the course, and I at once read very carefully Gödel's *1931* paper in the *Monatshefte*.

This paper contains Gödel's celebrated proof of the existence of undecidable sentences in formal systems embodying the usual elementary number theory, and his "second theorem" on the impossibility of a proof of the consistency of such a system within the system itself. Church's immediate reaction was that his formal system, about which I am going to say more, is sufficiently different from the systems Gödel dealt with that Gödel's second theorem might not apply to it (see Church *1933* top p. 843). Indeed, Church was right! In his system there is a proof of its own consistency, since in fact it is inconsistent (so all its sentences are provable), as Church had thought possible (*1933* top p. 842) and as Rosser and I showed later (Kleene and Rosser *1935*).

I wrote about Gödel's *1931* paper and his work generally in *1976*. In *1931* Gödel employed as a tool a

[1] A date shown in italics refers to a work listed in the References (under the name of the adjacent author).

class of number-theoretic functions, which he called "recursive" and which since my *1936* paper on general recursive functions have instead been called "primitive recursive".

Let me summarize at this point what was known about recursive functions in 1931, although it was not until later that I read more about them.

What are familiar as Peano's five axioms for the positive integers {1, 2, 3, ...} appeared in his *1889* and *1891*. As a companion to the fifth of these axioms, mathematical induction, he used definition by induction—in fact, primitive recursion (so called since Péter *1934*). Peano's axioms indeed come from Dedekind *1888*, who proved the theorem that a primitive recursion defines a function on the positive integers and applied it to the definition of the functions $m + n$, $m \times n$, and m^n. For agreement with the currently more usual setting for this theory, we can transpose from the positive integers {1, 2, 3, ...} to the globally more convenient natural numbers {0, 1, 2, ...} (used, for example, by Gödel *1931*).

The richness of the possibilities for the development of number theory on this basis was brought out by Skolem in his *1923* paper on the foundation of elementary arithmetic through the recursive mode of thought. Some of the devices used by Gödel *1931* with primitive recursive functions were anticipated therein.

Hilbert *1926* made a bold attempt to prove Cantor's continuum hypothesis by using recursions of more and more complicated kinds to generate the number-theoretic functions and by associating them with increasing ordinals of the second number class. The attempt failed, although something of its method survived in Gödel's proof of the consistency of the continuum hypothesis (*1938*, *1939*, *1940*). In the course of this attempt, Hilbert used an example of a function definable by a transfinite recursion (or by a recursion on two variables simultaneously) for which Ackermann had a proof (published in *1928*) that it is not primitive recursive.

This is what existed in 1931. Immediately thereafter, it was elaborated and extensively developed by Péter in a series of papers (*1932*, *1934*, *1935*, *1936*, etc.) depicting a hierarchy of levels of recursive functions: the primitive recursive functions, the double recursive functions, the triple recursive functions, and so on. The theory is expounded in her book *1951*.

Besides this theory of special recursive functions, for which the basis had been laid by 1931, there had existed since antiquity (for example, in Euclid's *Elements*, written about 330–320 B.C.) examples of algorithms as methods of deciding questions (predicates) or computing functions. The name "algorithm" is a corruption of the name of the ninth-century Arabian mathematician Al-Khowarizmi. (I participated in a scientific pilgrimage in September 1979 to his reputed birthplace at the Khowarizm oasis in Uzbekistan.)

As I said, in the fall semester of 1931–32 I was taking Church's course in which, besides reading Gödel *1931*, I was made acquainted with Church's postulates for the foundation of logic. I shall not go into these in full. They included one ingredient that has proved to be extraordinarily fruitful.

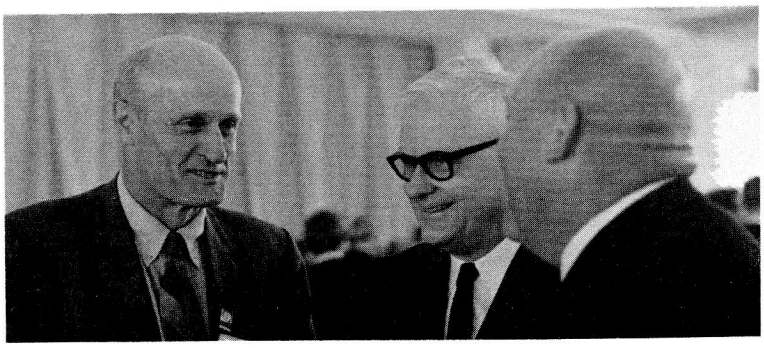
As calculus is usually taught, the undergraduate mathematics student is introduced to functions, some of which have been given closed names, permanent such as "sine" or temporary such as "f"; others are named by expressions, such as "$x^4 + 3x^2 + 2$" containing a variable "x", that tell us what the value of the function is for each value of that variable as argument. The calculus student is unlikely to worry whether the expression "$x^4 + 3x^2 + 2$" really denotes a number (a different one for each number x that "x" denotes) or a function. That there is an ambiguity here (indeed, "$x^4 + 3x^2 + 2$" is called the "ambiguous-value" notation for the said function) may be illustrated by comparing two statements: "$x^4 + 3x^2 + 2$ is not less than 2" and "$x^4 + 3x^2 + 2$ is even" (a function f is *even* iff, for all x, $f(-x) = f(x)$). The first statement can be read as about any one of the numbers that are the values of the function. The second statement is about the function. (Mathematicians have been accustomed to stating definitions using "if" where "if and only if" is meant. Recently, an elegant way of making their writing accurate has won acceptance: spell such an "if" as "iff".)

The ambiguity can be banished by adopting "$\lambda x[x^4 + 3x^2 + 2]$" as a notation, with "$x$" now bound by the prefix "λx", for the function itself.

Once this notation is introduced, there are three obvious operations with it. (For simplicity, I usually omit the quotation marks from now on.) First, if $f = \lambda x[x^4 + 3x^2 + 2]$, then $f(2) = 2^4 + 3(2^2) + 2 = 30$. So,

Note: The photographs in this article are from the author's personal collection.

Stephen C. Kleene was born in Hartford, Connecticut, in 1909. He graduated from Amherst College in 1930 and received a Ph.D. from Princeton University in 1934. He has taught mathematics at the University of Wisconsin since 1935. He was president of the Association for Symbolic Logic from 1956 to 1958 and was editor of the Journal of Symbolic Logic *from 1950 to 1962. He was elected to the National Academy of Sciences in 1969.*



Stephen C. Kleene (left) and Alonzo Church (center) with Anatol Ivanovic Malcev (back to camera) at a reception in Moscow in 1966.

without introducing f, $\{\lambda x[x^4 + 3x^2 + 2]\}(2)$ *reduces* (equivalently) to $2^4 + 3(2^2) + 2$. Second, inversely, $2^4 + 3(2^2) + 2$ *expands* (equivalently) to $\{\lambda x[x^4 + 3x^2 + 2]\}(2)$ (or indeed also to various other expressions, such as $\{\lambda x[x^4 + 3x^2 + x]\}(2)$). Third, there is the usual principle that a bound variable can be changed to any other with the same range, so long as (in complicated situations) there is no "collision". Thus our expression $\lambda x[x^4 + 3x^2 + 2]$ means the same as the expression $\lambda y[y^4 + 3y^2 + 2]$.

The device of this λ-notation with its operations is so simple that one wonders why in beginning calculus courses some bright student did not think of it.

Church introduced the λ-notation in the context of his system of postulates. In my illustration $\lambda x[x^4 + 3x^2 + 2]$, the context is the arithemtic of the real numbers, where there are available some constants (such as 2 and 3) and some functions already symbolized (such as sum and product). This brings us to the observation that, to make the "λ-calculus" precise, we need to describe in advance the class of the meaningful expressions in which we use it.

Let us be bold enough to do so in a minimal language, with just one category of variables (without type or sort), infinite in supply, constituting the atoms. We get this from Church's formalism by stripping away all the other elements, which for this purpose do not need to be present. (In my *1934* I got the same effect by treating the constants no differently than the variables, calling them all "proper symbols".)

The meaningful formulas we get in this way I call "λ-formulas". First, the variables are λ-*formulas*. Second, if P is a λ-*formula* already built up containing x as a free variable, λx[P] is also a λ-*formula*. (There is also a version of this theory, called the "λ-*K*-calculus", in which P is not required to contain x free.) I assume my readers understand free and bound occurrences of variables, where for us λx is the only operator that binds variables. I could build in the definition of this simultaneously with that of the class of the λ-*formulas*. We think of λx[P] as denoting that function of x whose value (if defined), for each value taken by x, is the value then taken by P. Third, if M and N are λ-*formulas*, so is {M}(N). We think of {M}(N) as the result of the application of a function M to an argument N. The λ-*formulas* are all and only the expressions that are such by repeated application of these three clauses. This is an example of an "inductive definition", a class of objects (in the present example, "λ-formulas") being defined to comprise all the objects that are required to be in it by the repeated application of certain clauses (here the three clauses identified by their opening words "First", "Second," and "Third"), and to comprise no other objects.

There are now the three operations by which λ-formulas can be transformed without altering their meanings. I shall state them in general. (I illustrated them above, but not in the present purified language.) First, there is *reduction* whereby {λx[P]}(N) is changed to the result $S^x_N P|$ of substituting N for the free occurrences of x in P, provided no "collision" of variables results. Second, there is *expansion*, the inverse of reduction. Third, there is the *change of a bound variable*, λx[P] becoming $\lambda y[S^x_y P|]$, again avoiding "collision". Each of these steps can be made on a whole λ-formula, or on a consecutive part that is a λ-formula, except that we do not apply expansion to the x of a prefix λx, as I noted in my *1934* emending Church's formulation.

As I have said, this and more are in Church's formulation of his language and postulates. We just leave out the more. Before research was done, no one guessed the richness of this subsystem. Who would have guessed that this formulation, generated as I have described to clarify the notation for functions, has implicit in it the notion (not known in mathematics in 1931 in a precise version) of all functions on the positive integers (or on the natural numbers) for which there are algorithms?

This requires that the positive integers (or the natural numbers) be identified within the class of the λ-formulas.

Before presenting Church's identification, I shall introduce some abbreviations, mainly his. λx[P] will often be written λx•P or simply λx P. {M}(N) when M is (or is named by) a single symbol can be simplified

to M(N). Moreover, λx[λy[P]] can be abbreviated λxy[P] or λxy•P or λxy P; {{M}(N)}(P) can be abbreviated {M}(N, P) or M(N, P). Herein is a workable definition by Schönfinkel *1924* (however, λxy[P] first appears in my *1934*) of two-place functions from one-place functions. A similar situation applies with more places (functions of p variables for $p > 2$).

For A and B λ-formulas, I now write "A red B" (read "A (is) reducible (to) B") to say that A is transformable to B by zero or more reductions of the above described form ({λx[P]}(N) to S^x_N P|, as a whole or part) with zero or more changes of bound variables. (This suits my present summary. At the time and in the literature, we were using instead "A conv B" (read "A (is) convertible (to) B") in which expansions are also allowed. In the case of the results taken over here from my writings, it will be clear from the constructions that they hold with "red" instead of "conv".)

Iff no reduction is possible on B, either immediately or after some changes of bound variables, B is said to be in *normal form*; and then, iff A red B, B is a *normal form of* A. (These notions were introduced in Church's 1931 lectures.)

Church identified the positive integers 1, 2, 3, ... with the following λ-formulas (in normal form):

$$(1) \qquad \lambda fx{\cdot}f(x), \lambda fx{\cdot}f(f(x)), \lambda fx{\cdot}f(f(f(x))), \ldots.$$

Here I am using the italic letters "f" and "x" as particular variables, while above the Roman "x" and "y" were names for any variables (distinct names standing for distinct variables).

Beginning in *1936a*, I instead identified the natural numbers 0, 1, 2, ... with the λ-formulas (1), partly because the primitive and general recursive functions I took over from Gödel were defined on the natural numbers. But here, while I am speaking of developments through 1935, it will be easier to stick in this respect with Church than to keep annotating the difference.

I call the λ-formulas (1) *numerals*, and specifically $\lambda fx{\cdot}f(x)$ *the numeral for* 1, $\lambda fx{\cdot}f(f(x))$ *the numeral for* 2, etc. When I use "n" to denote a positive integer, I use "$\boldsymbol{n}$" to denote the corresponding numeral.

Now we cannot escape the following. Each λ-formula F containing no free variables has an interpretation as a partial one-place function ϕ from the positive integers to positive integers, as I will indicate in a moment. A "partial" function ϕ is one such that, for each positive integer n, $\phi(n)$ either is defined with a positive integer as its value or is undefined.

"Partial" functions were first used explicitly in recursion theory in my *1938* (using the natural numbers). What I am stating now was formulated in 1931–32 as a condition when a λ-formula defines an ordinary or "total" one-place function ϕ from the positive integers to positive integers.

The partial function ϕ that a λ-formula F containing no free variables expresses is the function ϕ such that, for each positive integer n, $\phi(n) = m$ or $\phi(n)$ is undefined, according to whether F($\boldsymbol{n}$) red $\boldsymbol{m}$ for some positive integer m or not. That this defines a partial (single-valued) function depends on there being at most one m (for a given F and n) such that F($\boldsymbol{n}$) red $\boldsymbol{m}$. At the time of which I am speaking, we felt assured of this by the interpretation of the λ-calculus or of Church's full system. Subsequently, the Church-Rosser theorem *1936* became available, whereby A red B with B in normal form can hold, with a given A, for at most one B apart from the choices of the bound variables in it. Thereby, with a given F and n, F($\boldsymbol{n}$) red $\boldsymbol{m}$ for at most one m.

A formula F with the property just described we say λ-*defines* the function ϕ, and we then say that ϕ is λ-*definable*. (This terminology was not published, although we had been working with the concept since 1931–32, until in Church *1936* and my *1936a*.) Similarly, with functions of more variables, taking $\phi(n_1, \ldots, n_p) = m$ iff F($\boldsymbol{n}_1, \ldots, \boldsymbol{n}_p$) red $\boldsymbol{m}$.

If F contains free variables, F($\boldsymbol{n}$) red $\boldsymbol{m}$ for no n and m, since a sequence of reductions leaves the set of the free variables unaltered. So such an F simply λ-defines the totally undefined function.

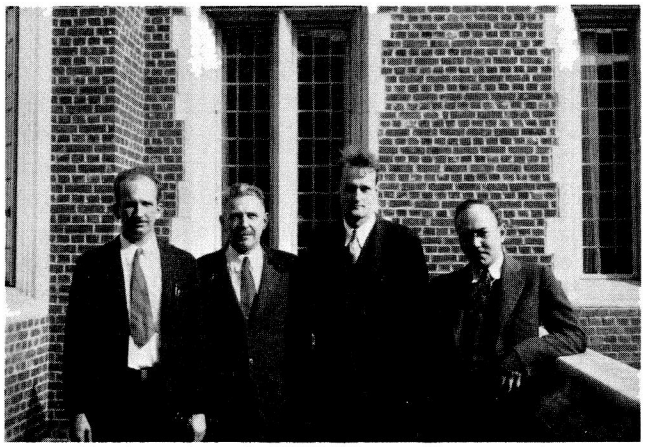
Church's particular identification (1) of λ-formulas with 1, 2, 3, ... was made with a view to facilitating definitions by induction. Thus, if F is the λ-formula $\lambda n{\cdot}n$(G, A), F($\boldsymbol{n}$) red or $n = 1, 2, 3, \ldots$ is reducible respectively to

$$(2) \qquad \mathrm{G(A)}, \mathrm{G(G(A))}, \mathrm{G(G(G(A)))}, \ldots.$$

We can say the sequence (2) of λ-formulas is λ-*defined* by this F. In the case A has a numeral $\boldsymbol{a}$ as its normal form, and G λ-defines a total function ψ, F λ-defines the function whose sequence of values is $\psi(a), \psi(\psi(a)), \psi(\psi(\psi(a))), \ldots$.

Church (*1933* p. 863, and in his lectures in 1931–32) gave two examples of λ-definable (total) functions. First, the successor function S (where $S(n) = n+1$) is λ-defined by $\lambda nfx{\cdot}f(n(f, x))$, call it "$S$". To illustrate for $n = 2$, we must verify that $S(2)$ red 3. After unabbreviating S and 2, we have
$\{\lambda nfx{\cdot}f(\{\{n\}(f)\}(x))\}(\lambda fx{\cdot}f(f(x)))$ red
$\lambda fx{\cdot}f(\{\{\lambda fx{\cdot}f(f(x))\}(f)\}(x))$ red
$\lambda fx{\cdot}f(\{\lambda x{\cdot}f(f(x))\}(x))$ red $\lambda fx{\cdot}f(f(f(x)))$, which is 3. Second, the sum function $m+n$ is λ-defined by $\lambda mn{\cdot}n(S, m)$ (abbreviate it "+"), as is easily seen.

In Church's postulates, means were provided to express propositional functions (predicates), and a descriptive operator ι such that ι(F) expresses "the x



Left to right: J. Barkley Rosser, Solomon Lefschetz, Stephen C. Kleene, and Edward J. McShane in Princeton in 1935.

such that F(x)". ι(λx[G]) is abbreviated ιx[G] or ιx•G. Having λ-defined S and +, Church did not continue to give λ-definitions of other functions, but instead he gave for subtraction − and product × the respective defining formulas $\lambda rs\cdot \iota x[\{+\}(x, s) = r]$ and $\lambda mn\cdot\{-\}(m(n(S),1),1)$, using his descriptive operator ι.

We can observe that all the λ-definable functions, in contrast to some of the functions that are definable using, for example, Church's ι-symbol and other constants, are "effectively calculable" or calculable by algorithms.

To justify this, I can appeal to the Church-Rosser theorem *1936*, which, beyond what I stated above, tells us that, if we get A red B with B in normal form by one series of reductions (with changes of bound variables), then any such series pursued long enough (or even any such series after any preliminary spree of using expansions or mixed expansions and reductions) must eventually lead to the same B, apart from alphabetical differences in the bound variables.

So, considering for the moment only the total functions, an algorithm for computing the function ϕ λ-defined by F is the following. For a given n, look for the part of F(***n***) of the form {λx[P]}(N) beginning leftmost and reduce it immediately, or immediately after changes of bound variables if necessary, and keep repeating. This makes the computation procedure determinate (apart from details in the choices of bound variables, which could also be made determinate by some conventions). We could vary it, without changing the fact of termination and the result, at any step where there is more than one part of the form {λx[P]}(N) by choosing to reduce one or another of those parts.

For partial functions, the "effective calculability" of ϕ means that there is an algorithm that leads in a finite number of steps to the value of $\phi(n)$ for any n for which the value is defined, and that for any other n leads to no value (either by terminating in a situation that does not give a value or else by continuing ad infinitum).

My study of the class of the λ-definable functions came about as follows. At the conclusion of Church's course in January 1932, I undertook as a Ph.D. thesis project to develop the theory of positive integers in the formal system of his *1932* and *1933*. Indeed, I took over the first part of the project proposed on the final page of his *1933*. This called at the outset for proving Peano's axioms in Church's formalism.

Proofs of three of them were "immediately evident", although I formulated the fifth axiom a little differently than Church (see my *1935* p. 157).

But the third axiom (if the successors of two numbers are equal, so are the numbers) offered a challenge. I proposed to handle it by defining the predecessor function P (where $P(1) = 1$, $P(S(n)) = n$) in the system, and indeed using only the λ-calculus.

Almost at once, I saw a way to change the identification of the integers with λ-formulas which made the predecessor function come out immediately. When I showed this to Church, he responded that my alternative identification would not do, because his was specifically chosen to give definition by induction (as indicated at (2) above, and illustrated by the λ-definition of +). I might have challenged this "put-down" by investigating whether in fact, with my alternative identification, definition by induction and further developments could not be managed reasonably well. I did not. The fact that I did not and the use of the λ-calculus instead of the λ-*K*-calculus in Church's system may have meant that I developed λ-definability in a more difficult but perhaps more challenging version. Thirty-one years later, in a letter dated January 20, 1963, Dana Scott communicated to me an alternative identification of the positive integers with λ-formulas making the predecessor function immediate (the same as, or similar to, mine of 1932, of which I preserved no record), and indicated success in the further development of the theory using that alternative. I am not sufficiently familiar with Scott's development and other recent work to be able to say as of 1979 what version of the theory is the best.

Soon after returning to Church's identification, one day late in January or early in February 1932, while in a dentist's office, it came to me that I could λ-define the predecessor function by using the λ-definability of (2) by $\lambda n\cdot n(\mathrm{G}, \mathrm{A})$ in the following way. The ordered number-triples (n_1, n_2, n_3) can be represented by the λ-formulas $\lambda fghx\cdot f(\ldots f(g(\ldots g(h(\ldots h(x)\ldots))\ldots))\ldots)$ with n_1 f's, n_2 g's, and n_3 h's after the prefix

$\lambda fghx$. And a λ-formula G is easily constructed to perform the following operation on any number triple:

$$(n_1, n_2, n_3)$$

$$\swarrow\swarrow\ \downarrow$$

$$(n_2, n_3, S(n_3)).$$

So if A is (1, 1, 1), then $\lambda n{\cdot}n$(G, A) λ-defines the sequence of number-triples

(3) (1, 1, 2), (1, 2, 3), (2, 3, 4), (3, 4, 5), ...

It is then easy by a λ-formula H to erase all but the first number of each triple so as to obtain

(4) 1, 1, 2, 3, ...,

which is the sequence of values of the predecessor function P. Thus P is λ-defined by $\lambda n{\cdot}$H(n(G, A)); call it "P". When I brought this result to Church, he told me that he had just about convinced himself that there is no λ-definition of the predecessor function.

The discovery that the predecessor function is after all λ-definable excited our interest in what functions are not just definable in the full system but actually λ-definable. The exploration of this became a major subproject for my Ph.D. thesis. Of course, I did develop a great deal of theory of positive integers in Church's formalism, using many λ-definitions in the process.

After my thesis was accepted (September 1933) and before it was published (*1935*), Rosser and I established that the full formal system of Church is inconsistent (as was suspected in the fall of 1933, finally established in the spring of 1934, and published in Kleene and Rosser *1935*). I then (in the spring of 1934) rewrote my thesis to retain, first, only as much of the theory developed in Church's full system as was used in the proof of its inconsistency, and second, the theory of λ-definability of functions, divorced from the inconsistent system and standing solidly on the Church-Rosser theorem. When it began to appear that the full system is inconsistent, Church spoke out on the significance of λ-definability, abstracted from any formal system of logic, as a notion of number theory. (I do not think this significance had been escaping me. On p. 23 of the manuscript of my thesis as submitted for publication on October 9, 1933, before its revision, there is a statement to the effect that all the formal definitions in the thesis are actually λ-definitions.)

Let me sketch how my subproject went, from February 1932 on. Church and I knew that only effectively calculable functions can be λ-definable. We kept thinking of specific such functions, and of specific operations for proceeding from such functions to others. I kept establishing the functions to be λ-definable and the operations to preserve λ-definability.



Haskell Curry with Bruce Kleene (the author's son) on his shoulders at the Kleene farm in Hope, Maine, in 1949.

Schönfinkel (*1924*) and Curry (*1929*, *1930*, *1932*) had developed combinatory logic, in which variables are not used, being replaced by constants that serve their functions. Rosser, for his Ph.D. thesis under Church (published *1935*) brought this into relation with the λ-calculus. In Rosser's version of combinatory logic, two constants I and J suffice, which translate into the λ-calculus as $\lambda x{\cdot}x$ and $\lambda fxyz{\cdot}f(x, f(z, y))$. Thus, in combinatory logic J(F, X, Y, Z) reduces (by itself, or as part of a larger expression) to F(X, F(Z, Y)), just as in the λ-calculus $\{\lambda fxyz{\cdot}f(x, f(z, y))\}$(F, X, Y, Z) does. Let us call the formulas we get in the λ-calculus using only variables, I, and J "combinations". Thus, a variable is a *combination*. I and J (as translated into the λ-calculus) are *combinations*. If M and N are *combinations*, so is {M}(N). The *combinations* are all and only the expressions that are such by repeated applications of these three clauses. The *terms* in a combination are the occurrences of variables, I, and J in it.

By Rosser's results (already known to me in the spring of 1932), for any λ-formula A there is a combination A′ such that A′ red A. (Actually, what I took over from Rosser was that A′ conv A. But without digging into Rosser's arguments, we can see that A′ red A, by induction on the construction of A. Take the case A is λxP. By the hypothesis of the induction, there is P′ such that P′ red P. Say P′ consists of the terms $t_1, \ldots, t_n$, each either x or some other variable, or one of I and J. Replace in P′ each of $t_1, \ldots, t_n$ that is an I or J by a different variable to get P″. By Rosser's result, there is a combination R″ such that

R″ conv λxP″. However, λxP″ is in normal form, so by the Church-Rosser theorem, R″ red λxP″. If we replace in this reduction each variable that in P″ represents a term I or J of P′ by the I or J represented, the sequence of reduction steps that worked with the variables will still work. Thus, making the replacements in R″ to get R′, we have a combination R′ with R′ red λxP′ red λxP. The basis and the induction step for {M}(N) are immediate.)

One evening in the spring term of 1932, while listening to a concert at Princeton, I saw how to use this to establish (inter alia) that, if A and B are any two λ-formulas with the same free variables, there is a λ-formula L such that L(1) red A and L(2) red B. The idea for the construction of L is first to replace A and B by combinations A′ and B′ with A′ red A and B′ red B. The terms in these combinations can be listed. The resulting double list of terms, but with each distinct variable listed only once, can be trussed up into a λ-formula L with the following property. When L is applied to the numeral 1, the terms coming from A′ are thrown into place in their original structure to give A, while the terms coming only from B (not variables) are disintegrated, and vice versa when applied to 2 (my *1934* pp. 537–538). This form of definition by cases was very useful.

To illustrate, consider the problem of λ-defining a function ϕ defined by a primitive recursion of the rudimentary form $\phi(1) = a$, $\phi(S(n)) = \psi(\phi(n))$. Supposing the function ψ can be λ-defined by G, we will want to λ-define a sequence of λ-formulas of the form

(5) A, G(A), G(G(A)), G(G(G(A))), ...

Of course, if A is reducible to a numeral, and G λ-defines a total function, A and G will contain no free variables. But in the theory of λ-definability, we can and do consider the λ-definability of this sequence under only the obvious condition that the free variables of G are among those of A. In brief, we want to prefix A to the sequence of λ-formulas (2). (If we had identified the positive integers, or the natural numbers, with $\lambda fx{\cdot}x$, $\lambda fx{\cdot}f(x)$, $\lambda fx{\cdot}f(f(x))$, ..., in the λ-K-calculus, (5) instead of (2) would have been immediate.) First, consider any λ-formula F with the same free variables as A. By cases, we can get a formula L such that L(1) red F and L(2) red $\lambda n{\cdot}n(I,$ A). Now take F′ to be $\lambda n{\cdot}$L$(n(P, 3), P(n))$. It is easily seen that, for $n = 1, 2, 3, \ldots$, F′$(\boldsymbol{n})$ reduces to

(6) A, F(1), F(2), F(3), ...

So, when F is $\lambda n{\cdot}n$(G, A), which λ-defines (2), F′ is the desired λ-formula λ-defining (5).

Now consider the general form of a primitive recursion for a function of one variable: $\phi(1) = a$, $\phi(S(n)) = \psi(n, \phi(n))$. Now we want

(7) A, G(1, A), G(2, G(1, A)), ...

In (5), let its A and G be $\lambda n{\cdot}n(I,$ A) and $\lambda rn{\cdot}$G$(n, r(P(n)))$, and for the F′ λ-defining (5) then let F″ be $\lambda n{\cdot}$F′$(n, P(n))$. It is not hard to see that F″ λ-defines (7).

Finally, in this series of illustrations, take a primitive recursion with a parameter: $\phi(1, x) = \alpha(x)$, $\phi(S(n), x) = \psi(n, \phi(n, x), x)$. We want a λ-formula F‴ such that, for $n = 1, 2, 3, \ldots$, and any x, F‴$(\boldsymbol{n}, \boldsymbol{x})$ reduces respectively to

(8) A$(\boldsymbol{x})$, G(1, A$(\boldsymbol{x})$, $\boldsymbol{x}$), G(2, G(1, A$(\boldsymbol{x})$, $\boldsymbol{x}$), $\boldsymbol{x}$), ...

In (7), let its A and G be A(x) and $\lambda nr{\cdot}$G(n, r, x), and take F‴ to be $\lambda nx{\cdot}$F″(n).

I also treated some variations of the schema of primitive recursion, which now we know from Péter *1934* or otherwise (see my *1952* Chapter IX) to be reducible to primitive recursion, and further, for example, double recursion.

In racking my brains for still more examples of effective definitions on which to try out my ability to λ-define, I thought of the least-number operator "the least y such that", which since my *1938* I have written "μy". If $R(x, y)$ is a total (i.e., completely defined) effectively decidable relation, and for each x there is a y such that $R(x, y)$, then $\mu yR(x, y)$ is a total function $\phi(x)$, which I considered to be effectively calculable, even if one can give in advance no bound for the y. (If a bound $\psi(x)$ for the y can be found for each x, the definition of $\phi(x)$ can be effected primitive-recursively from $R(x, y)$ and $\psi(x)$, as was known to Skolem in *1923* and Gödel in *1931*.) Otherwise, but with $R(x, y)$ effectively decidable, $\mu yR(x, y)$ is an effectively calculable partial function, as will appear in our treatment.

Frequently, descriptive definitions of functions of positive integers can be expressed in terms of the μ-operator.

To treat $\mu yR(x, y)$ in the λ-calculus, say the relation $R(x, y)$ is represented by a λ-formula R with R$(\boldsymbol{x}, \boldsymbol{y})$ red 2 if $R(x, y)$ and R$(\boldsymbol{x}, \boldsymbol{y})$ red 1 if not $R(x, y)$. I constructed a λ-formula $\wp$ such that, if D$(\boldsymbol{k})$ red 2, then $\wp$(D, $\boldsymbol{k}$) red $\boldsymbol{k}$, and if D$(\boldsymbol{k})$ red 1, then $\wp$(D, $\boldsymbol{k}$) red $\wp$(D, $\boldsymbol{k+1}$). So $\wp$(R$(\boldsymbol{x})$, 0) red $\boldsymbol{y}$ for the least y such that R$(\boldsymbol{x}, \boldsymbol{y})$ red 2 and each of R$(\boldsymbol{x}$, 0), ..., R$(\boldsymbol{x}, \boldsymbol{y-1})$ red 1. Thus λx$\wp$(R(x), 0) λ-defines $\mu\, yR(x, y)$. The letter "$\wp$" stood for "perpetual motion function"; if D$(\boldsymbol{i})$ red 1 for all i, then $\wp$(D, 0) red $\wp$(D, 1) red $\wp$(D, 2) ... ad infinitum.

This result greatly extended our horizon for rendering effective definitions by λ-definitions. Also, it became clear at once that a host of particular problems of elementary number theory can be subsumed under the problem of determining whether any given λ-formula C has a normal form (see Kleene *1935* pp. 232–233, Church *1936* pp. 358–359). In a sense, this observation set the stage for the undecidability proof by Church *1936* for this problem, and the same for the halting problem for Turing machines (see my *1952* p. 382).

Subsequently, after writing and rewriting my thesis and before July 1, 1935, I thought of what in *1952* I called the "recursion theorem" (*1938, 1952* pp. 352–353) and established it in the λ-calculus, calling it "circular definition" (*1936a*, (19) p. 346 and discussion p. 347): For any λ-formula G and any positive integer p, there is a λ-formula F such that, for every $x_1, \ldots, x_p$,

$$\{F\}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p) \text{ red } \{G\}(F, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_p),$$

and, if G contains no free variables, a λ-formula H such that H(F) red I, which enables us to erase F when it is in the way (for example, when G represents a definition by cases with F not used in all cases).

The recursion theorem treats an absolutely general form of recursion. If a function ϕ is defined by saying that any value $\phi(x_1, \ldots, x_p)$ shall be obtained by operating on the arguments $x_1, \ldots, x_p$ and the function ϕ itself by a given functional ψ, and ψ is λ-definable in the obvious sense for functionals, the theorem tells us that then ϕ is λ-definable. In general, a function ϕ so defined is partial. The theorem separates the issue of whether ϕ is λ-definable (it is) from the issue of what p-tuples of numbers it is defined for (at worst, none). Actually, G here need not λ-define a functional ψ, but $\{G\}(F, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_p)$ may be reducible to a numeral depending on what λ-formula F is, not just on what partial function it λ-defines.

In my *1936a*, λ-definability theory is reworked from the beginning to obtain definition by cases and the recursion theorem quickly, and primitive recursion and the least-number operator follow as applications of the recursion theorem, which I used in 1951 in studying von Neumann's self-reproducing automata.

Let us back up a little in time to 1933, to see how λ-definability related to other developments. The concept of λ-definability existed full-fledged by the fall of 1933 and was circulating among the logicians at Princeton. Church had been speculating, and finally definitely proposed, that the λ-definable functions are all the effectively calculable functions—what he published in *1936*, and which I in *1952* Chapter XII (or almost in *1943*) called "Church's thesis".



Kurt Gödel (right) and his wife, Adele (second from left), with the author's parents, Gustav A. Kleene and Alice C. Kleene, in Hope, Maine, in 1941.

When Church proposed this thesis, I sat down to disprove it by diagonalizing out of the class of the λ-definable functions. But, quickly realizing that the diagonalization cannot be done effectively, I became overnight a supporter of the thesis.

Gödel came to the Institute for Advanced Study in the fall of 1933. According to a November 29, 1935, letter from Church to me, Gödel "regarded as thoroughly unsatisfactory" Church's proposal to use λ-definability as a definition of effective calculability. Church "replied that if [Gödel] would propose any definition of effective calculability which seemed even partially satisfactory [Church] would undertake to prove that it was included in lambda-definability."

Soon thereafter, in his lectures in the spring of 1934, Gödel took a suggestion that had been made to him by Herbrand in a letter in 1931 and modified it to secure effectiveness. The result was what is now known as "Herbrand-Gödel general recursiveness".

Herbrand's suggestion (as reported in Gödel *1934*) was this: "If ϕ denotes an unknown function, and $\psi_1, \ldots, \psi_k$ are known functions, and if the ψ's and the ϕ are substituted in one another in the most general fashions and certain pairs of the resulting expressions are equated, then if the resulting set of functional equations has one and only one solution for ϕ, ϕ is a recursive function."

Gödel's modification consisted, besides in being a bit more specific about the form of the equations, in

requiring that, for each set of natural numbers $x_1, \ldots, x_p$ as arguments of ϕ, exactly one equation of the form $\phi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p) = \boldsymbol{m}$ (with numerals in appropriate symbolism) is deducible by a substitution rule and a replacement rule from the set of functional equations and the equations giving the values of $\psi_1, \ldots, \psi_k$, presumed to be previously defined in a similar manner.

In a February 15, 1965, letter to Martin Davis, Gödel wrote, "However, I was, at the time of these lectures [*1934*], not at all convinced that my concept of recursion comprises all possible recursions . . .".

I somewhat edited the details in my *1936* and *1943* (see also my *1952* pp. 274–275), in particular wrapping together all of the equations defining all of $\psi_1, \ldots, \psi_k, \phi$, back to the starting point, into one system E of equations as the recursive definition of ϕ.

Church (*1936*) and I (*1936a*) published equivalence proofs for Herbrand-Gödel general recursiveness to λ-definability. So, under Church's thesis, there were now two exact mathematical characterizations of the intuitive notion of all effectively calculable functions, or all functions for which algorithms exist in the sense exemplified by many particular examples in over two millennia of mathematical history.

People have asked me how I thought of the normal form theorem for general recursive functions that I gave in my *1936*. (From *1936* on, my writing was in terms of the natural numbers rather than the positive integers.) The theorem includes that each general recursive function is obtainable using only primitive recursions (with explicit definitions) and the least-number operator (used just once).

I had been preconditioned by my work on λ-definability to think in terms of these elements. Thus I had confirmed in my Ph.D. thesis that all primitive recursions (as well as explicit definitions) can be effected in the λ-calculus, and likewise the least-number operator. Part of the project for my *1936a* paper was to prove that every general recursive function is λ-definable; so I could not help but reflect that I could do that if I could get every general recursive function by a combination of primitive recursions (with explicit definitions) and least-number operations.

With this motivation, I came up with an idea akin to the following. First, we can represent the stages in the computation of a value of a general recursive function ϕ (here, the deductions from the system E of equations defining ϕ recursively) by Gödel numbers, and characterize the numbers representing such stages (deductions) primitive-recursively. I knew the uses of Gödel numbering very well from my study of Gödel *1931*. Second, we can search by the least-number operator for the first number y that represents a stage recognizable primitive-recursively as terminal for the computation of the value of ϕ for given arguments $x_1, \ldots, x_p$. Third, having found such a y, from it we can extract primitive-recursively the value m. Moreover, in this process, the system E of equations defining ϕ recursively is represented by its Gödel number e. Using these numbers as parameters, each general recursive function ϕ is expressible in the form

$$\phi(x_1, \ldots, x_p) = U(\mu y T(e, x_1, \ldots, x_p, y))$$

for some natural number e, where U is a fixed primitive recursive function, and T is a fixed primitive recursive predicate (a $(p+2)$-ary relation). This summarizes what seems the best version of the proof (Kleene *1943*). In this version, the method used applies directly to any of the known characterizations of the effectively calculable functions—for example, to λ-definability and to Turing computability. (In *1936*, instead of characterizing primitive-recursively the Gödel numbers of deductions, I enumerated primitive-recursively the Gödel numbers of deducible equations. The idea of a "recursively enumerable class [set]" first appeared there.)

That the equation just exhibited holds, for a given general recursive function ϕ, for some e (a Gödel number of E), for all $x_1, \ldots, x_p$, depends on the foregoing formulation whereby a system E of equations defines ϕ recursively iff, for each $x_1, \ldots, x_p$, there is exactly one m for which an equation of the form $\phi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p) = \boldsymbol{m}$ is deducible from E.

It remained for me in *1938* (the work was done in 1936) to omit this assumption about E, and just talk about those p-tuples $x_1, \ldots, x_p$ for which there is an m, still assumed to be unique, such that $\phi(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p) = \boldsymbol{m}$ is deducible from E. So $\phi(x_1, \ldots, x_p)$ becomes a partial function. The functions so definable I called *partial recursive*, and for them the equation of the normal form theorem becomes

$$\phi(x_1, \ldots, x_p) \simeq U(\mu y T(e, x_1, \ldots, x_p, y))$$

where "$\simeq$" means that the two members are either both defined with the same value or both undefined. This generalization (should I say "partialization"?) of the notion of general recursive function had been positively avoided by Church *1936* and Turing *1936-37*, *1937* (see my *1978* footnote 1).

In my *1938* application of partial recursive functions, I needed the recursion theorem. Remembering that I had it in the λ-calculus, I found a proof of it less than one line long in my theory of partial recursive functions. In my *1938* footnote 7 and *1952* p. 340, I introduced the notation "$\{e\}(x_1, \ldots, x_p)$" for the right side

of the normal form theorem (just above) as a partial recursive function of $e, x_1, \ldots, x_p$. Using this notation, the statement of the recursion theorem here exactly parallels my statement of it above for the λ-calculus: For any natural number g and positive integer p, there is a natural number f such that, for every $x_1, \ldots, x_p$,

$$\{f\}(x_1, \ldots, x_p) \simeq \{g\}(f, x_1, \ldots, x_p).$$

This works because in the theory of partial recursive functions any natural number, like g or f here, represents a partial (indeed, partial recursive) function. Just as with G in the λ-calculus, what g gives on the right side may depend on f itself and not just on what partial function f "defines recursively". This was useful in the applications.

The last of the original three equivalent exact definitions of effective calculability is computability by a Turing machine. I assume my readers are familiar with the concept of a Turing machine, after Turing *1936–37*.

Turing learned of the work at Princeton on λ-definability and general recursiveness just as he was ready to send off his manuscript, to which he then added an appendix outlining a proof of the equivalence of his computability to λ-definability. In *1937* he gave a proof of the equivalence in detail. Post's short note *1936*, containing the same idea as Turing's paper, was independent of Turing's work but not of the work at Princeton.

Turing *1936–37* was concerned primarily with machines that perform the continuing computation (ad infinitum) of infinite sequences of 0's and 1's printed on alternate squares of the machine tape (once printed not being erased or changed), while performing scratch work (subject to erasures and changes) on the intervening squares. A *computable* real number (between 0 and 1) is one whose binary expansion can be so computed.

Turing then defined a one-place number-theoretic function ϕ to be *computable* iff there is a machine that computes the sequence of 0's and 1's with $\phi(0)$ 1's before the first 0 and, for $x > 0$, $\phi(x)$ 1's between the xth and the $(x+1)$st 0. He remarked that a similar definition can be given of computable functions of several variables.

While fully honoring Turing's conception of what his machines could do, I was skeptical that his was the easiest way to apply them to the computation of number-theoretic functions. In any case, only a total function $\phi(x)$ can be computed in his way. Hence, in lectures at Madison, Wisconsin, in the spring of 1941 (and in my *1952* Chapter XIII), I applied his machines differently, more like in Post *1936*.

First, I stipulated that any natural number n shall be represented in computation by a block of $n+1$ tallies printed on consecutive squares of the machine tape, preceded and followed by a blank square. A p-tuple of natural numbers $x_1, \ldots, x_p$ is then represented by p blocks of $x_1+1, \ldots, x_p+1$ tallies, respectively, with a blank square before the first, between each two, and after the last, block.

I then said that a partial function $\phi(x_1, \ldots, x_p)$ is *Turing computable* iff some Turing machine, queried (as I shall explain) with any p-tuple $x_1, \ldots, x_p$ of natural numbers, answers with the value $\phi(x_1, \ldots, x_p)$ if that value is defined, and otherwise does not answer. We query the machine with $x_1, \ldots, x_p$ by presenting to it the p-tuple $x_1, \ldots, x_p$ represented as above on its tape (assumed to be infinite to the right) with the tape otherwise blank, with the machine scanning the rightmost tally of the representation, in what I called its "first active state". Machine states are what Turing called "m-configurations", and I supposed a given machine to have a (finite) list of "active" states (from which the machine performs an act or in Turing's terminology a "move") and one "passive" state. The machine, so queried, answers that $\phi(x_1, \ldots, x_p) = m$ iff at some later moment of time it reaches the passive state (stops) with the $(p+1)$-tuple $x_1, \ldots, x_p, m$ represented on its tape and the rightmost tally of that representation scanned. Because of the determinateness of each successive act, this can happen (for given $x_1, \ldots, x_p$) for at most one m.

The equivalence proofs to λ-definability and recursiveness in Turing *1937* together with those in my *1952* Chapter XIII established the equivalence of this version of Turing computability to Turing's in *1936–37* when ϕ is total; otherwise his version does not apply. For one who would work directly from Turing *1936–37*, a useful critique is provided by the appendix to Post *1947*.

For rendering the identification with effective calculability the most plausible—indeed, I believe compelling—Turing computability has the advantage of aiming directly at the goal, as is clear (and as Turing modestly suggested in *1937* p. 153).

It seems that only after Turing's formulation appeared did Gödel accept Church's thesis, which had then become the Church-Turing thesis. In the Postcriptum to the Davis *1965* reprint of his *1934*, Gödel wrote, "Turing's work gives an analysis of the concept of 'mechanical procedure' (alias 'algorithm' or 'computation procedure' or 'finite combinatorial procedure'). This concept is shown to be equivalent with that of a 'Turing machine'." In a conversation at San Juan on October 31, 1979, Davis expressed to me the opinion that the equivalence between Gödel's defini-

tion of general recursiveness and mine (which equivalence Gödel, in his February 15, 1965, letter to Davis, called "not quite trivial"), and my normal form theorem, were considerations that combined with Turing's arguments to convince Gödel of the Church-Turing thesis.

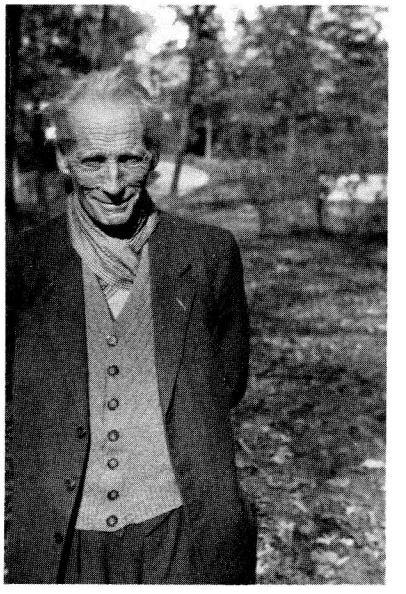
Because Turing's formulation works directly with machines, albeit idealized ones (error-free, and with potentially infinite memory), I think it must have considerable practical significance, although I have not been closely concerned with this. Turing himself went into computing at the National Physical Laboratory in 1945–48 and from 1948 on at the Computing Machine Laboratory at Manchester, England.

The earliest notion, λ-definability, has (as I have related) the remarkable feature that it is all contained in a very simple and almost inevitable formulation, arising in a natural connection with no prethought of the result. And a given λ-formula engenders the computation procedure for the function it defines. Of course, the λ-formula may be complicated. Under Herbrand-Gödel general recursiveness, and my partial recursiveness adapted from it, one works with systems E of equations that can be very unwieldy. Under Turing computability one may have very long machine tables. Indeed, Turing *1937* p. 153 spoke of the λ-definitions as "more convenient". (As I see it, convenience for one or another purpose requires testing in practice.)

I myself, perhaps unduly influenced by rather chilly receptions from audiences around 1933–35 to disquisitions on λ-definability, chose, after general recursiveness had appeared, to put my work in that format. (I did later publish one paper *1962* on λ-definability in higher recursion theory.) I thought general recursiveness came the closest to traditional mathematics. It spoke in a language familiar to mathematicians, extending the theory of special recursiveness, which derived from formulations of Dedekind and Peano in the mainstream of mathematics.

I cannot complain about my audiences after 1935, although whether the improvement came from switching I do not know. In retrospect, I now feel it was too bad I did not keep active in λ-definability as well. So I am glad that interest in λ-definability has revived, as illustrated by Dana Scott's 1963 communication.

My normal form theorem gives a means of working with general (or partial) recursive functions that gets one away from contemplation of the usually complicated system E of equations. This provides a formulation, "μ-recursiveness" (*1952* p. 320), in which one can phrase much of the theory conveniently, irrespective of what original formulation one started with.



L. E. J. Brouwer in Madison, Wisconsin, in 1953.

Subsequently, various other equivalents of the three notions that arose in the mid-1930's have appeared. Their sponsors claim that these equivalents have considerable merits. (Without having worked in them, I cannot assess their merits.) I mention Post's formulation using his "canonical systems" (*1943*), Markov's "algorithms" (*1951*, *1954*) (akin to Post's *1943*, but corresponding to partial, instead of general, recursiveness), and a formulation of Smullyan using his "elementary formal systems" (*1961*) (also akin to Post's *1943*).

My survey, thus far, has taken us through the basic elements of the theory of recursive functions of positive integers or of natural numbers. Much more has happened in recursion theory.

With the emergence of the notion of general recursive function, the preexisting notions of special recursiveness (for example, Péter *1936*) gave a subrecursive hierarchy. Other ways of getting subrecursive hierarchies have been studied since then. I have barely touched the area (my *1958* with Axt's *1959*, *1963*), and I do not consider myself qualified to survey it.

The first unsolvability or undecidability results were in Church *1936* and *1936a*, my *1936* and *1943*, and Turing *1936–37*. Subsequent work, beginning with Post *1947* and Markov *1947*, has established the unsolvability of a variety of mathematical decision problems "not on their face specially related to logic" (as Church expressed to me what he hoped would happen on a postcard dated May 19, 1936), in algebra, topology, and real-variable analysis. References may be found in my *1967* p. 264 and Boone *1968*.

The intuitionistic school of mathematics, founded by Brouwer (*1908*, *1918–19*, etc.), accepted only mathematical proofs that are "constructive". In the spring of 1941 I conjectured that this should mean that an intuitionistic proof of a proposition of the form "for all x, there exists a y such that $R(x, y)$" should implicitly determine a general recursive function ϕ such that, for all x, $R(x, \phi(x))$. I confirmed this in my *1945* taken with Nelson's *1947* (exposition in my *1952* § 82). Similar results were obtained for intuitionistic analysis (Kleene and Vesley *1965*). Survey in my *1973*.

Church and Kleene *1936*, Church *1938*, and Kleene *1938* applied the notions of λ-definability and recursiveness to characterizing effectiveness in defining transfinite ordinals. Thus arose a theory of constructive ordinals, further investigated in my *1955a* with *1944*, and in Spector *1955*.

Recursiveness relativized to a class of number-theoretic problems, or, as I would apply it, to a total number-theoretic predicate or to a set of numbers or to a total number-theoretic function, was introduced by Turing in his *1939* (written at Princeton), with his suggestive imagery of an "oracle". For example, with a total function ψ, a function ϕ is (*general or partial*) *recursive in* ψ, iff ϕ is computed by a machine like Turing's *1936–37* machines except for having access to an oracle, who, asked "What is the value of $\psi(x)$?" for any x that comes up in the course of the machine's calculations, will always answer with the correct value. Following Gödel *1934*, let us represent a predicate $P(x)$ by the function $\phi(x)$ with $\phi(x) = 0$ when $P(x)$ is true and $\phi(x) = 1$ when $P(x)$ is false. Then we say P is *recursive* iff ϕ is. Similarly, representing $Q(x)$ by $\psi(x)$, P is *recursive in* Q iff ϕ is recursive in ψ.

Post in *1948* defined "degrees of unsolvability" of predicates, sets, or functions. "Unsolvability" suggests that he is dealing (primarily) with predicates, etc., for which there is no algorithm or solution to the "decision problem" for whether the predicate holds for given arguments or whether a given number belongs to the set or to the "computation problem" of finding an algorithm to compute the function. A predicate P is of the *same degree* as Q, iff P is recursive (after Turing *1939*) in Q and vice versa; of *lower degree* than Q, iff P is recursive in Q but not vice versa. A *degree* is the set of all the predicates, sets, and functions that have the same degree as a given one of them. The lowest "degree of unsolvability", called "solvability", consists of the general recursive predicates, sets, and functions. In *1944* I joined Post in studying the structure of the system of his degrees. An extensive field of research grew out of this (for example, Sacks *1963*, Shoenfield *1971*).

Post *1944* dealt with recursively enumerable sets. By including as such not just the sets enumerated by a general recursion function (Kleene *1936*) but also the empty set, he made the recursively enumerable sets coincide with the sets S for which the predicate $a \in S$ is expressible in the form $(Ex)R(a, x)$ with R recursive, which I discuss below. He proposed there the problem, which (after his *1948*) we can state thus: Are all recursively enumerable but nonrecursive sets of the same degree of unsolvability? An affirmative answer would mean that all proofs of unsolvability of decision problems for formal systems (or for recursively enumerable sets) can in principle be established by "reducing" the decision problem in hand to one particular unsolvable such problem. "Post's problem", as this came to be known, resisted solution until 1956. Friedberg (*1956*, *1957*), then a 20-year-old senior at Harvard, and Muchnik (*1956*, *1958*), in Russia, of similar age, independently proved that there exist pairs of recursively enumerable sets of incomparable degrees of unsolvability, answering Post's question negatively.

In *1943* I introduced a hierarchy of predicates, subsequently called the "arithmetical hierarchy", obtained by starting with the recursive predicates and prefixing more and more quantifiers ("for all x" or "(x)"; "(there) exists (an) x (such that)" or "(Ex)"). This gives a classification of the predicates used in elementary number theory (or "arithmetic"). Indeed, consider the predicate forms

$$(Ex)R(a, x) \quad (x)(Ey)R(a, x, y) \quad \ldots$$

$$R(a)$$

$$(x)R(a, x) \quad (Ex)(y)R(a, x, y) \quad \ldots$$

where the R in each stands for a general recursive predicate (equivalently after the first, a primitive recursive predicate). To each of the forms after the first, there is a predicate of the variable a expressible in that form, but not in the other form with the same number of quantifiers, nor in any of the forms with fewer quantifiers. This result (obtained in 1940) marks the beginning of the use of applications and adaptations of recursive function theory to reveal structure in parts of classical mathematics where effectiveness does not in general obtain. The farther up in this arithmetical hierarchy one must go to define a predicate, the higher is its degree in the sense of Post *1948*. By Kleene and Post *1954*, Post's degrees give a fine structure of differing degrees within each level of the arithmetical hierarchy above the lowest. Mostowski *1947* obtained the arithmetical hierarchy independently, in a little different format in which he gave it

as analogous to the hierarchy of projective sets in descriptive set theory. Addison *1954*, *1955*, *1958*, *1960* investigated this analogy and others.

By considering Turing's computation by a machine having access to an oracle, but, with the rules governing the machine (including how it puts questions to the oracle and what it does when the oracle gives any answer) fixed, varying the oracle so that she answers for one or another value of a one-place function variable α, I obtained in *1950* the notion of a general recursive function with a function variable. In brief, $\phi(a)$ can be regarded as a *recursive* function $\phi(a, \alpha)$ of two variables, iff $\phi(a)$ is recursive in α (by a Turing oracle-machine) uniformly in α. We can omit the number variable here, or, in general, we can have any finite number of variables of each type, number and one-place function. (I did this then for my studies of intuitionism.)

In *1955*, I used this to build a hierarchy of number-theoretic predicates of a by applying quantifiers with function variables to arithmetical predicates: the "analytic hierarchy". The arithmetical hierarchy constitutes the lowest level in the analytic hierarchy, just as the general recursive predicates constitute the lowest level in the arithmetical hierarchy. The arithmetical hierarchy can alternatively be extended, less steeply, to transfinite levels (indexed by constructive ordinals of the first and second number classes). In effect, we use transfinite successions of number quantifiers. This gives the "hyperarithmetical hierarchy". In *1955b*, I showed that the predicates falling in this hierarchy are exactly those that are expressible in both one-quantifier forms of the analytic hierarchy. This is the analog of the result of my *1943*, Post *1944*, and Mostowski *1947* that a predicate is general recursive exactly if it is expressible in both one-quantifier forms of the arithmetical hierarchy.

Addison initiated discussions at Madison and at Warsaw that led to the proposal of the notations Σ_k^j, Π_k^j, which are now standard for these hierarchies (Addison *1958*, Mostowski *1959*). Here Σ_k^0 is the class of the sets S for which the predicate $a \in S$ is expressible (or of the predicates expressible) in the k-quantifier form of the arithmetical (or hyperarithmetical) hierarchy with an existential quantifier first; Π_k^0, with a universal quantifier first. Σ_k^1, Π_k^1 relate similarly to the analytic hierarchy; for $j > 1$, Σ_k^j, Π_k^j to the hierarchies with quantifiers of higher finite types which I predicted in *1955* p. 312 and *1955b* p. 212 and studied in *1959* and *1963*. The further notation $\Delta_k^j = \Sigma_k^j \cap \Pi_k^j$ came into use in the late 1960's. Thereby, the two theorems just stated can be written concisely:

$$\text{hyperarithmetical} = \Delta_1^1 \text{ and general recursive} = \Delta_0^1.$$

Currently, research is in progress on a theory using recursive functions of any finite types 0, 1, 2, ... of variables. Type 0 is the natural numbers, and type $j+1$ is the one-place functions from type j to type 0. This theory was opened up by my *1959* and *1963* (survey in Kechris-Moschovakis *1977*). Also, the "inductive definitions", of which many examples were used in the foregoing developments, have become a subject of study in general (for example, Spector *1961*, Moschovakis *1974*).

My invitation to lecture at the FOCS meeting mentioned interest in my work published as "Representation of events in nerve nets and finite automata" *1956* (and 1951). About the genesis of that, I can say little more than that I had the luck to find myself again working in a context in which significant developments were implicit but not yet explicit—just as in 1931–32 when I was given Church's formalism to study, in which the λ-calculus was maybe implicit or maybe explicit, but all that it developed to be was certainly only implicit.

In the summer of 1951, as a visitor at the RAND Corporation in Santa Monica (through an invitation obtained by my old friend of Princeton days, Merrill Flood), I was given the McCulloch and Pitts paper *1943*, with their mathematical model for nerve nets, to see what I would make of it. I found their model to be very interesting—an original contribution—but their analysis of it to fall quite short of what was possible. So I did what cried out to be done with it (as it seemed to me), having newly in mind also the idea of a finite automaton, which came to me that summer at RAND through reading in printer's proof (?) von Neumann's Hixon Symposium lecture *1951*. My work in this area was all done in a few months in 1951 at Santa Monica and back at Madison, and a bit of time in 1955 in editing the resulting Project RAND Research Memorandum for *Automata Studies*. This work was an interpolation into the midst of a very pressing program of research in recursive function theory, broadly construed. I had then much work long in progress but not yet in final form. Therefore, I felt compelled to invest my further time where I already had a large investment to protect. This was painful to me, because I could see many alluring problems opening up in the field of automata theory that I had just touched.

Herewith I conclude my testimony.

Acknowledgment

The preparation of this paper was supported in part by the National Science Foundation under Grant No. MCS79-01439.

REFERENCES

Ackermann, Wilhelm

1928. Zum Hilbertschen Aufbau der reellen Zahlen. *Mathematische Annalen 99,* 118–133 (English trans. in van Heijenoort, 1967, pp. 493–507).

Addison, John W.

1954. *On Some Points of the Theory of Recursive Functions.* Ph.D. dissertation, University of Wisconsin.

1955. Analogies in the Borel, Luzin, and Kleene hierarchies, I and II. *Abstracts, Bul. Amer. Math. Soc. 61,* 75 and 171–172.

1958. Separation principles in the hierarchies of classical and effective descriptive set theory. *Fundamenta Mathematicae 46,* 123–135.

1960. The theory of hierarchies. *Logic, Methodology and Philosophy of Science. Proc. of 1960 International Congress.* E. Nagel, P. Suppes, and A. Tarski, eds. Stanford, Stanford University Press, 1962, pp. 26–37.

Axt, Paul

1959. On a subrecursive hierarchy and primitive recursive degrees. *Trans. Amer. Math. Soc. 92,* 85–105.

1963. Enumeration and the Grzegorczyk hierarchy. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik 9,* 53–65.

Boone, William W.

1968. Decision problems about algebraic and logical systems as a whole and recursively enumerable degrees of unsolvability. *Contributions to Mathematical Logic.* K. Schütte, ed. Amsterdam, North-Holland, pp. 13–33, 72–74.

Brouwer, L. E. J.

1908. De onbetrouwbaarheid der logische principes (The untrustworthiness of the principles of logic). *Tijdschrift voor Wijsbegeerte 2,* 152–158.

1918–19. Begründung der Mengenlehre unabhängig vom logischen Satz vom ausgeschlossenen Dritten. *Verhandelingen der Koninklijke Akademie van Wetenschappen te Amsterdam (Eerste sectie) 12,* 5 (1918), 43 pp.; 7 (1919) 33 pp.

Church, Alonzo

1932. A set of postulates for the foundation of logic. *Annals of Math., 2s. 33,* 346–366.

1933. A set of postulates for the foundation of logic (second paper). *Annals of Math., 2s. 34,* 839–864.

1936. An unsolvable problem of elementary number theory. *Amer. J. Math. 58,* 345–363.

1936a. A note on the Entscheidungsproblem. *J. Symbolic Logic 1,* 40–41. Correction, 101–102.

1938. The constructive second number class. *Bul. Amer. Math. Soc. 44,* 224–232.

Church, Alonzo, and S. C. Kleene

1936. Formal definitions in the theory of ordinal numbers. *Fundamenta Mathematicae 28,* 11–21.

Church, Alonzo, and J. B. Rosser

1936. Some properties of conversion. *Trans. Amer. Math. Soc. 39,* 472–482.

Curry, Haskell B.

1929. An analysis of logical substitution. *Amer. J. Math. 51,* 363–384.

1930. Grundlagen der kombinatorischen Logik. *Amer. J. Math. 52,* 509–536, 789–834.

1932. Some additions to the theory of combinators. *Amer. J. Math. 54,* 551–558.

Davis, Martin

1965. *The Undecidable. Basic Papers on Undecidable Propositions, Unsolvable Problems and Computable Functions.* Hewlett, N.Y., Raven Press, 440 pp.

Dedekind, Richard

1888. *Was sind und was sollen die Zahlen?* Braunschweig (English trans. in Dedekind, *Essays on the Theory of Numbers.* Chicago, Open Court, 1901, 29–115).

Friedberg, Richard M.

1956. [Article concerning him.] *Time 67,* 12 (March 19, 1956), 83.

1957 (abstract 1956). Two recursively enumerable sets of incomparable degrees of unsolvability (solution of Post's problem, 1944). *Proc. Nat. Acad. Sci. 43,* 236–238. Abstract, *Bul. Amer. Math. Soc. 62* (1956), 260.

Gödel, Kurt

1931. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I. *Monatshefte für Mathematik und Physik 38,* 173–198 (English trans. in Davis, 1965, 4–38, and in van Heijenoort, 1967, 592–616).

1931–32a. [Remarks in] Diskussion zur Grundlegung der Mathematik. *Erkenntnis 2,* 147–148.

1934. *On undecidable propositions of formal mathematical systems.* Mimeographed notes by S. C. Kleene and J. B. Rosser on lectures at the Institute for Advanced Study, 1934, 30 pp. (reprinted in Davis, 1965, 39–74).

1938. The consistency of the axiom of choice and of the generalized continuum-hypothesis. *Proc. Nat. Acad. Sci. 24,* 556–557.

1939. Consistency-proof for the generalized continuum-hypothesis. *Proc. Nat. Acad. Sci. 25,* 220–224.

1940. *The consistency of the axiom of choice and of the generalized continuum-hypothesis with the axioms of set theory.* Notes by George W. Brown on lectures at the Institute for Advanced Study, 1938–39. *Annals of Mathematics Studies 3,* Princeton, Princeton University Press, 66 pp.

Hilbert, David

1926. Über das Unendliche. *Mathematische Annalen 95,* 161–190 (English trans. in van Heijenoort, 1967, 367–392).

Kechris, Alexander S., and Yiannis N. Moschovakis

1977. Recursion in higher types. *Handbook of Mathematical Logic.* Jon Barwise, ed. Amsterdam, North-Holland, pp. 681–737.

Kleene, Stephen C.

1934. Proof by cases in formal logic. *Annals of Math., 2s. 35,* 529–544.

1935. A theory of positive integers in formal logic. *Amer. J. Math. 57,* 153–173, 219–244.

1936. General recursive functions of natural numbers. *Mathematische Annalen 112,* 727–742.

1936a. λ-definability and recursiveness. *Duke Math. J. 2,* 340–353.

Kleene, Stephen C. (*continued*)
1938. On notation for ordinal numbers. *J. Symbolic Logic 3,* 150–155.
1943. Recursive predicates and quantifiers. *Trans. Amer. Math. Soc. 53,* 41–73.
1944. On the forms of the predicates in the theory of constructive ordinals. *Amer. J. Math. 66,* 41–58.
1945. On the interpretation of intuitionistic number theory. *J. Symbolic Logic 10,* 109–124.
1950. Recursive functions and intuitionistic mathematics. *Proc. Int. Congress of Mathematicians, Cambridge, Mass.* Providence, Amer. Math. Soc., 1952, I, 679–685.
1952. *Introduction to Metamathematics.* Amsterdam, North-Holland, xi+550 pp. (eighth reprint 1980).
1955. Arithmetical predicates and function quantifiers. *Trans. Amer. Math. Soc. 79,* 312–340.
1955a. On the forms of the predicates in the theory of constructive ordinals (second paper). *Amer. J. Math. 77,* 405–428.
1955b. Hierarchies of number-theoretic predicates. *Bul. Amer. Math. Soc. 61,* 193–213.
1956. Representation of events in nerve nets and finite automata. *Automata Studies.* C. E. Shannon and J. McCarthy, eds. *Annals of Mathematics Studies 34,* 3–41 (slightly altered from Project RAND Research Memorandum RM-704, 15 December 1951, 101 pp.).
1958. Extension of an effectively generated class of functions by enumeration. *Colloquium Mathematicum 6,* 67–78.
1959. Recursive functionals and quantifiers of finite types I. *Trans. Amer. Math. Soc. 91,* 1–52.
1962. Lambda-definable functionals of finite types. *Fundamenta Mathematicae 50,* 281–303.
1963. Recursive functionals and quantifiers of finite types II. *Trans. Amer. Math. Soc. 108,* 106–142.
1967. *Mathematical Logic.* New York, John Wiley & Sons, xiii+398 pp.
1973. Realizability: a retrospective survey. *Cambridge Summer School in Mathematical Logic, 1971.* A. R. D. Mathias and H. Rogers, eds. *Lecture Notes in Mathematics 337,* Berlin, Springer-Verlag, 95–112.
1976. The work of Kurt Gödel. *J. Symbolic Logic 41,* 761–778. An addendum, *43* (1978), 613.
1978. Recursive functionals and quantifiers of finite types revisited I. *Generalized recursion theory II, Proc. 1977 Oslo Symposium.* J. E. Fenstad, R. O. Gandy, and G. Sacks, eds. Amsterdam, North-Holland, 185–222.

Kleene, S. C., and Emil L. Post
1954. The upper semi-lattice of degrees of recursive unsolvability. *Annals of Math., 2s. 59,* 379–407.

Kleene, S. C., and J. B. Rosser
1935. The inconsistency of certain formal logics. *Annals of Math., 2s. 36,* 630–636.

Kleene, S. C., and R. E. Vesley
1965. *The Foundations of Intuitionistic Mathematics.* Amsterdam, North-Holland, viii+206 pp.

Markov, A. A.
1947. On the impossibility of certain algorithms in the theory of associative systems (A. A. Markoff). *Comptes rendus (Doklady) de l'Académie des Sciences de l'URSS, n.s. 55,* 583–586 (English trans. of Russian original).
1951. Theory of algorithms. *Amer. Math. Soc. Translations, 2s. 15* (1960), 1–14 (English trans. of Russian original).
1954. *Theory of Algorithms.* National Science Foundation, U.S. Department of Commerce, and Israel Program for Scientific Translation (1961), vi+444 pp. (English trans. of Russian original).

McCulloch, Warren S., and Walter Pitts
1943. A logical calculus of the ideas immanent in nervous activity. *Bul. Math. Biophysics 5,* 115–133.

Moschovakis, Yiannis N.
1974. *Elementary Induction on Abstract Structures.* Amsterdam, North-Holland, 174 pp.

Mostowski, Andrzej
1947. On definable sets of positive integers. *Fundamenta Mathematicae 34,* 81–112.
1959. On various degrees of constructivism. *Constructivity in Mathematics. Proc. of Colloquium, Amsterdam, 1957,* A. Heyting, ed., Amsterdam, North-Holland, 178–194.

Muchnik, A. A.
1956. Nerazrešimosť problemy svodimosti teorii algoritmov (Negative answer to the problem of reducibility of the theory of algorithms). *Doklady Akademii Nauk S.S.S.R., n.s. 108,* 194–197.
1958. Solution of Post's reduction problem and some other problems of the theory of algorithms. I. *Amer. Math. Soc. Translations, 2s. 29* (1963), 197–215 (English trans. of Russian original).

Nelson, David
1947. Recursive functions and intuitionistic number theory. *Trans. Amer. Math. Soc. 61,* 307–368.

Peano, Giuseppe.
1889. *Arithmetices Principia, Nova Methodo Exposita.* Turin, Bocca, xvi+20 pp. (English trans. in van Heijenoort, 1967, 83–97).
1891. Sul concetto di numero. *Rivista di Matematica 1,* 87–102, 256–267.

Péter, Rózsa
1932. Rekursive Funktionen. *Verhandlungen des Internationalen Mathematiker-Kongresses Zürich 2,* 336–337.
1934. Über den Zusammenhang der verschiedenen Begriffe der rekursiven Funktion. *Mathematische Annalen 110,* 612–632.
1935. Konstruktion nichtrekursiver Funktionen. *Mathematische Annalen 111,* 42–60.
1936. Über die mehrfache Rekursion. *Mathematische Annalen 113,* 489–527.
1951. *Rekursive Funktionen.* Budapest, Akadémiai Kiadó (Akademischer Verlag), 206 pp. *Recursive Functions,* Third revised edition, New York, Academic Press, 1967, 300 pp.

Post, Emil L.
1936. Finite combinatory processes—formulation 1. *J. Symbolic Logic 1,* 103–105.

1943. Formal reductions of the general combinatorial decision problem. *Amer. J. Math. 65,* 197–215.

1944. Recursively enumerable sets of positive integers and their decision problems. *Bul. Amer. Math. Soc. 50,* 284–316.

1947. Recursive unsolvability of a problem of Thue. *J. Symbolic Logic 12,* 1–11.

1948. Degrees of recursive unsolvability. Abstract (Preliminary report). *Bul. Amer. Math. Soc. 54,* 641–642.

Rosser, J. Barkley

1935. A mathematical logic without variables. *Annals of Math., 2s. 36,* 127–150; *Duke Math. J. 1,* 328–355.

Russell, Bertrand

1919. *Introduction to Mathematical Philosophy.* London, Geo. Allen and Unwin; New York, Macmillan, viii+208 pp.

Sacks, Gerald E.

1963. *Degrees of unsolvability. Annals of Mathematics Studies 55,* ix+174 pp.

Schönfinkel, Moses

1924. Über die Bausteine der mathematischen Logik. *Mathematische Annalen 92,* 305–316 (English trans. in van Heijenoort, 1967, 355–366).

Shoenfield, Joseph R.

1971. *Degrees of Unsolvability.* Amsterdam, North-Holland, viii+111 pp.

Skolem, Thoralf

1923. Begründung der elementaren Arithmetik durch die rekurrierende Denkweise ohne Anwendung scheinbare Veränderlichen mit unendlichem Ausdehnungsbereich. *Skrifter utgit av Videnskapsselskapet i Kristiania, I. Matematisk-Naturvidenskabelig Klasse 1923 6,* 38 pp. (English trans. in van Heijenoort, 1967, 302–333).

Smullyan, Raymond M.

1961. *Theory of formal systems. Annals of Mathematics Studies 47,* xi+142 pp.

Spector, Clifford

1955. Recursive well-orderings. *J. Sym. Logic 20,* 151–163.

1961. Inductively defined sets of natural numbers. *Infinitistic methods, Proc. Symposium on Foundations of Mathematics, Warsaw, 1959,* Oxford, Pergamon; Warsaw, Państwowe Wydawnictwo Naukowe, pp. 97–102.

Turing, Alan M.

1936–37. On computable numbers, with an application to the Entscheidungsproblem. *Proc. London Math. Soc., 2s. 42,* 230–265. A correction, *43* (1937), 544–546.

1937. Computability and λ-definability. *J. Symbolic Logic 2,* 153–163.

1939. Systems of logic based on ordinals. *Proc. London Math. Soc., 2s. 45,* 161–228.

van Heijenoort, Jean

1967. *From Frege to Gödel: a Source Book in Mathematical Logic, 1879–1931.* Cambridge, Harvard University Press, xi+660 pp.

von Neumann, John

1951. The general and logical theory of automata. *Cerebral Mechanisms in Behavior: The Hixon Symposium, September 1948, Pasadena.* L. A. Jeffress, ed. New York, John Wiley & Sons, pp. 1–31.

Whitehead, Alfred North

1911. *An Introduction to Mathematics.* London, Williams and Norgate; New York, Henry Holt, vi+256 pp.