THE BULLETIN OF SYMBOLIC LOGIC
Volume 1, Number 1, March 1995

THE MATHEMATICAL WORK OF S. C. KLEENE

J. R. SHOENFIELD

§1. The origins of recursion theory. In dedicating a book to Steve Kleene, I referred to him as the person who made recursion theory into a theory. Recursion theory was begun by Kleene's teacher at Princeton, Alonzo Church, who first defined the class of recursive functions; first maintained that this class was the class of computable functions (a claim which has come to be known as Church's Thesis); and first used this fact to solve negatively some classical problems on the existence of algorithms. However, it was Kleene who, in his thesis and in his subsequent attempts to convince himself of Church's Thesis, developed a general theory of the behavior of the recursive functions. He continued to develop this theory and extend it to new situations throughout his mathematical career. Indeed, all of the research which he did had a close relationship to recursive functions.

Church's Thesis arose in an accidental way. In his investigations of a system of logic which he had invented, Church became interested in a class of functions which he called the λ-definable functions. Initially, Church knew that the successor function and the addition function were λ-definable, but not much else. During 1932, Kleene gradually showed[1] that this class of functions was quite extensive; and these results became an important part of his thesis *1935a* (completed in June of 1933).

It was almost immediate from the definition that every λ-definable function was computable, i.e., that for each such function there was an algorithm by which, given an argument to the function, we could compute the corresponding value. As Kleene's investigations continued, Church came to believe that all computable functions were λ-definable. Kleene himself was doubtful; at first he tried to disprove it by a use of the diagonal method.

In 1934, Gödel, already the world's most famous logician, came to the Institute for Advanced Study. Church proposed his thesis to Gödel, who was very skeptical. A little later, Gödel defined another class of functions,

Received September 22, 1994.
We are grateful to the University of Wisconsin for the picture on the opposite page.

[1]Kleene writes that he achieved his first success in this direction, the proof that the predecessor function was λ-definable, while having some wisdom teeth pulled. How many wisdom teeth? In one place he says two; in another, four. I am afraid that this number is not computable at this stage.

1079-8986/95/0101-0003/$01.90

each of which was clearly computable; but he was by no means convinced at the time that all computable functions were in this class. Eventually, Church and Kleene *1936b* proved that Gödel's class was identical to the class of λ-definable functions; the members of this class came to be known as the *recursive* functions.

Kleene was finally convinced of Church's Thesis when he discovered the Recursion Theorem (discussed below). About this time, Alan Turing, unaware of the work being done at Princeton, had defined a class of functions which he claimed was the class of computable functions. In his definition, the class of functions was the class of all functions which could be computed by a particular kind of computing machine. He gave arguments that his machines could compute any computable function; these arguments were very convincing to Gödel and others. Eventually Turing read of Church's work and came to Princeton; while there, he published a proof [13] that his class of functions was equal to the class of recursive functions. From this point, Church's Thesis was accepted by almost all logicians.

One result of all this was the virtual disappearance of λ-definability from recursion theory. Kleene found, as he had remarked, that λ-definability did not have much sex appeal for logicians; so he switched to a variant of Gödel's definition. More recently, λ-definability has been revived and studied by computer scientists. A good introduction to the modern theory of λ-definability is [3].

Church realized at an early stage that his thesis made it possible to prove mathematically that finding algorithms to solve certain problems was impossible. He combined this idea and techniques used by Gödel in proving the Incompleteness Theorem[2] to show that there was no algorithm for deciding, given a sentence of the predicate calculus, if it was a theorem. A bit later, Turing showed that there was no algorithm for deciding, given an input for his machine, if the machine would eventually stop if started with this input. Since then, many difficult and interesting results on the nonexistence of algorithms, such as the unsolvability of Hilbert's tenth problem and of the word problem for groups, have been proved; and these results are one of the most important justifications of recursion theory.

§**2. Basic recursion theory.** Kleene began the development of basic recursion theory in *1936a*, and extended and improved it in *1952b*. We are going to sketch the principal features of this development. From now on, everything we do is attributable to Kleene unless otherwise indicated.

[2]Kleene has pointed out that if Gödel had not already proved his Incompleteness Theorem, Church might well have derived it from his results. The reader interested in such a derivation of the Incompleteness Theorem can consult my book ***Mathematical Logic***. Of course, this takes nothing away from Gödel's supreme achievement in discovering the Incompleteness Theorem.

First we consider which functions to allow. Our functions will take natural numbers (i.e., nonnegative integers) as arguments and values. They can be n-ary for any n. An n-ary function need not include among its arguments all n-tuples of numbers; if it does, it is called a *total* function. (In classical recursion theory, including all of Kleene's work, *function* means *total function* and what we have called a function is called a *partial function*. Partial functions were not even used in recursion theory until Kleene introduced them in *1938*. However, they have replaced total functions in most of recursion theory, which explains our change of terminology.) A total unary function is called a *real*. (This terminology from set theory is not yet popular in recursion theory, but it should be.)

We also want to consider relations; specifically, n-ary relations between natural numbers. (Thus a unary relation is just a set of numbers.) In recursion theory, it is convenient to identify a relation R with its *representing function*, which is the total function F defined by $F(x_1, \dots, x_n) = 0$ if $R(x_1, \dots, x_n)$ and $F(x_1, \dots, x_n) = 1$ otherwise. We shall understand *number* to mean *natural number* and *set* to mean *set of numbers*. (Other sets will be referred to as classes.) We use x, y, and z for numbers; F, G, and H for functions; P, Q, and R for relations; and A, B, and C for sets.

Next we need a definition of recursive. We have already mentioned three definitions, due to Church, Gödel, and Turing; we shall use a fourth, due to Kleene. We first list some simple properties which we want the class of recursive functions to have. We shall write $\vec{x}$ for $x_1, \dots, x_n$ when it is not important to indicate n.

I. The functions Z, Sc, and I defined by $Z(x) = 0$, $Sc(x) = x + 1$, and $I(x) = x$ are recursive.

II. If G is recursive and F is defined by[3] $F(x_1, \dots, x_n) = G(x_{i_1}, \dots, x_{i_k})$ where $1 \leq i_j \leq n$ for all j, then F is recursive.

III. If G and H are recursive and F is defined by $F(\vec{x}) = G(H(\vec{x}), \vec{x})$, then F is recursive.

IV. If G and H are recursive and F is defined inductively by $F(0, \vec{x}) = G(\vec{x})$, $F(y + 1, \vec{x}) = H(F(y, \vec{x}), y, \vec{x})$, then F is recursive.[4]

V. If G is recursive and F is defined by $F(\vec{x}) = \mu y(G(y, \vec{x}) = 0)$, then F is recursive. (Here $\mu y(G(y, \vec{x}) = 0)$ is equal to z if $G(z, \vec{x}) = 0$ and $G(y, \vec{x})$ is defined and different from 0 for every $y < z$. If there is no such z, $\mu y(G(y, \vec{x}) = 0)$ is undefined. In the important case when G is total, we may read μy as *the least y such that*.)

---

[3]When we write an equality between expressions which may be undefined, we mean that either both sides are undefined, or both sides are defined and have the same value.

[4]As Kleene pointed out in *1936c*, we can use an idea of Gödel to leave out IV, provided we add the functions $+$, $\cdot$ and $=$ to I. While this is useful for some purposes, it would make the following development more complicated.

We say a class of functions $\mathcal{A}$ is *recursively closed* if I–V are true when ‘is recursive’ is replaced by ‘is in $\mathcal{A}$’. A *recursive* function is a function which belongs to every recursively closed class. Since the intersection of any number of recursively closed classes is recursively closed, the class of recursive functions is the smallest recursively closed class; so I–V are correct as written.

A function F is *computable* if there is an algorithm which, if applied to the inputs $\vec{x}$, will give the output $F(\vec{x})$ if $F(\vec{x})$ is defined and will give no output otherwise. It is easy to see that the class of computable functions is recursively closed; so every recursive function is computable. The converse is by no means evident; in fact, there is no reason to believe it at this stage of the development.

From I–V we can prove that many functions and relations are recursive, e.g., all constant total functions, the functions $+$ and $\cdot$, and the relations $=, <, >, \leq$ and $\geq$. We can also prove results of the same nature as I–V, of which we mention two.[5]

VI. If P and Q are recursive relations, then $\neg P, P \vee Q, P \,\&\, Q, P \rightarrow Q, P \leftrightarrow Q, P_1$, and P_2 are recursive, where P_1 and P_2 are defined by $P_1(z, \vec{x}) \leftrightarrow (\forall y < z)P(y, z, \vec{x})$ and $P_2(z, \vec{x}) \leftrightarrow (\exists y < z)P(y, z, \vec{x})$.

VII. Let $R_1, \dots, R_k$ be recursive relations such that for all $\vec{x}$, exactly one of $R_1(\vec{x}), \dots, R_k(\vec{x})$ holds. If $G_1, \dots, G_k$ are recursive and total, and F is defined by

$$F(\vec{x}) = G_i(\vec{x}) \text{ if } R_i(\vec{x})$$

for $i = 1, \dots, k$, then F is recursive.

These consequences are fairly easy to prove; we note only that the P_1 and P_2 in VI should be defined by induction on z.

How shall we treat functions whose arguments and values are objects other than numbers? One method (first introduced by Gödel in his proof of the Incompleteness Theorem) is *coding*. To each of these objects we assign a number, called the *code* of that object, in such a way that different objects have different codes. If we have a function with these objects as arguments or values, we replace each such argument or value by its code. (The term ‘code’ was not used by Kleene, but it has become standard in recent years.)

Here is an important example of coding. Let the objects be all finite sequences of natural numbers. Let $p_0, p_1, \dots$ be the primes in increasing order. Then the code of the sequence $x_1, \dots, x_n$ is $\prod_{i=1}^{n} p_i^{x_i+1}$, which we designate by $\langle x_1, \dots, x_n \rangle$. A useful property of this coding is that $x_i < \langle x_1, \dots, x_n \rangle$; we call this the *sequence code inequality*.

[5]The fact that these results could be derived from I–V (or even I–IV) was noted by Gödel in his proof of the Incompleteness Theorem.

In this and other codings, the exact choice of codes is not important. What is important is that certain functions and relations associated with the coding are recursive. For example, $\langle x_1, \dots, x_n \rangle$ is a recursive function of $x_1, \dots, x_n$. The set of codes of sequences is recursive. There are recursive total functions lh and $(x)_i$ such that if $x = \langle x_0, \dots, x_{n-1} \rangle$, then $lh(x) = n$ and $(x)_i = x_i$ for $i < n$. Note that the statement that every member of the sequence with code x satisfies P can be written as $(\forall i < lh(x))P((x)_i)$.

One of the uses of codes of sequences is to replace n-ary functions by unary functions. If F is n-ary, we define the *contraction* $\langle F \rangle$ of F by $\langle F \rangle(x) = F((x)_0, \dots, (x)_{n-1})$. Then $F(x_1, \dots, x_n) = \langle F \rangle(\langle x_1, \dots, x_n \rangle)$. These equations show that F is recursive iff $\langle F \rangle$ is recursive.

Another use of codes of sequences is to give an extension of IV. In a definition under IV, the value of $F(y, \vec{x})$ depends on the value of $F(y - 1, \vec{x})$. In a more general form of definition by induction (sometimes called definition by *course-of-values induction*), the value of $F(y, \vec{x})$ depends on the value of $F(z, \vec{x})$ for all $z < y$. To code these values by a single number, we define $\overline{F}(y, \vec{x}) = \langle F(0, \vec{x}), \dots, F(y-1, \vec{x}) \rangle$. Then a definition by course-of-values induction has the form $F(y, \vec{x}) = G(\overline{F}(y, \vec{x}), y, \vec{x})$. We want to show that if G is recursive, then F is recursive. Since $F(y, \vec{x}) = (\overline{F}(Sc(y), \vec{x}))_y$, it is enough to prove that $\overline{F}$ is recursive. But it is easy to see that $\overline{F}$ has an inductive definition which comes under IV.

We now assign numbers, called *indices*, to recursive functions. We assign the index $\langle 1, 0 \rangle$ to Z, $\langle 1, 1 \rangle$ to S, and $\langle 1, 2 \rangle$ to I. If F and G are as in II and g is an index of G, then $\langle 2, g, \langle n, i_1, \dots, i_p \rangle \rangle$ is an index of F. If F, G, H are as in III and g, h are indices of G, H, then $\langle 3, g, h \rangle$ is an index of F. If F, G, H are as in IV and g, h are indices of G, H, then $\langle 4, g, h \rangle$ is an index of F. If F, G are as in V and g is an index of G, then $\langle 5, g \rangle$ is an index of F. Clearly no number is an index of two recursive functions, and every recursive function has at least one index. It is easy to prove that the set of indices is recursive by giving a definition by course-of-values induction, using the sequence code inequality.

We let $\{f\}(x_1, \dots, x_n)$ be $F(x_1, \dots, x_n)$ where f is an index of F; if f is not the index of an n-ary function, then $\{f\}(x_1, \dots, x_n)$ is undefined. By a *formula*, we mean an equation of the form $\{f\}(x_1, \dots, x_n) = y$ where $f, x_1, \dots, x_n, y$ are particular numbers and f is the code of an n-ary function. The code of this formula is $\langle f, \langle x_1, \dots, x_n \rangle, y \rangle$. The set of codes of formulas is recursive.

We shall show that the true formulas are the theorems of a formal system. The axioms of the formal system are the true formulas $\{f\}(\vec{x}) = y$ where f is the code of a function in I. There are four rules of inference, one for each of II–V; we give the last two and leave the other two to the reader. The rule corresponding to IV says: $\{\langle 4, g, h \rangle\}(0, \vec{x}) = z$ is inferable from $\{g\}(\vec{x}) = z$, and $\{\langle 4, g, h\rangle\}(y+1, \vec{x}) = z$ is inferable from $\{\langle 4, g, h\rangle\}(y, \vec{x}) = v$ and $\{h\}(v, y, \vec{x}) = z$. The rule corresponding to V says: $\{\langle 5, g\rangle\}(\vec{x}) = z$ is inferable from $\{g\}(z, \vec{x}) = 0$ and $\{g\}(y, \vec{x}) = v_y$ for all $y < z$, provided that each $v_y \neq 0$. As usual, a *proof* is a nonempty finite sequence of formulas, each of which is inferable from earlier formulas in the sequence (where an axiom is said to be inferable from the empty set); and a *theorem* is a formula which is the last formula in some proof.

Since formulas inferable from true formulas are true, every theorem is true. It is easy to prove that every true formula $\{f\}(\vec{x}) = y$ is a theorem by induction on f, using the sequence code inequality.

The code of a proof is the code of the sequence consisting of the codes of the formulas in the proof. The set of codes of proofs is recursive. The proof of this requires writing out all the rules of inference; it is tedious but straightforward.

We let $T_n(f, x_1, \dots, x_n, y)$ mean that y is the code of a proof, and that the last formula of the proof has $\{f\}(x_1, \dots, x_n)$ as its left side; this is the famous Kleene T-predicate. Each T_n is recursive. It is easy to define a recursive real U such that if y is the code of a proof, then $U(y)$ is the number which appears on the right side of the last formula in the proof. We then have

$$\{f\}(x_1, \dots, x_n) = U(\mu y T_n(f, x_1, \dots, x_n, y)).$$

This important equality is called the Normal Form Theorem.

If F is $(n+1)$-ary, we define $F_{(y)}$ by $F_{(y)}(\vec{x}) = F(y, \vec{x})$. We say that F *enumerates* a class of functions if that class has $F_{(0)}, F_{(1)}, \dots$ as its members. Thus if $F(f, x_1, \dots, x_n) = \{f\}(x_1, \dots, x_n)$, then F enumerates the class of recursive n-ary functions. It is an immediate consequence of the Normal Form Theorem that this F is recursive. This result is called the Enumeration Theorem. (Note that it is trivial that $\{f\}(x_1, \dots, x_n)$ is a recursive function of $x_1, \dots, x_n$ for each f; but this is a much weaker statement than the Enumeration Theorem.)

If F is recursive, then each $F_{(y)}$ is recursive; for $F_{(y)}(\vec{x}) = F(G(\vec{x}), \vec{x})$ where G is the constant function with value y. By examining the assignment of indices to recursive functions, we can define a total recursive function S such that if f is an index of F, then $S(f, y)$ is an index of $F_{(y)}$. We can then define S_m inductively so that $S_m(f, y_1, \dots, y_m)$ is an index of the function $F_{(y_1)\cdots(y_m)}$.

We now show that in defining a recursive function F, we may use an index of F.

Recursion Theorem. *If G is recursive, there is a recursive function F with an index f such that $F(\vec{x}) = G(f, \vec{x})$ for all $\vec{x}$.*

To prove this, define a recursive function H by $H(y, \vec{x}) = G(S(y, y), \vec{x})$, and let h be an index of H. Let $F = H_{(h)}$ and $f = S(h, h)$. Then F is recursive; f is an index of F; and $F(\vec{x}) = H(h, \vec{x}) = G(S(h, h), \vec{x}) = G(f, \vec{x})$.

The Recursion Theorem has many uses in recursion theory. We shall show how it can be used to prove that certain functions are recursive. If G and H are total, then $F(0, x) = G(x)$, $F(y + 1, x) = H(F(y, 2x), y, x)$ is a legitimate definition of $F(y, x)$ by induction on y; but it does not come under IV. We want to show that if G and H are recursive, then F is recursive. Define K by

$$\begin{aligned} K(f, 0, x) &= G(x), \\ K(f, y + 1, x) &= H(\{f\}(y, 2x), y, x). \end{aligned}$$

By VII and the Enumeration Theorem, K is recursive. Hence there is a recursive F' with an index f such that $F'(y, x) = K(f, y, x)$. This F' satisfies the equations for F; so $F' = F$ and hence F is recursive.

Now let us sketch how we prove that our definition of recursive is equivalent to other definitions. Suppose $\mathcal{A}$ is the class of functions given by the alternative definition. First we prove that $\mathcal{A}$ is recursively closed. Then every recursive function is in $\mathcal{A}$. Now let F be in $\mathcal{A}$. Generally, the alternative definition will provide a method of computing $F(\vec{x})$. After suitably coding the objects occurring in the computation, we can show by the above methods that the output of the computation is a recursive function of the inputs. But this means that F is recursive.

What kind of evidence does the theory give for Church's Thesis? We can hope that, given an algorithm for computing F, we can prove that F is recursive by the above methods. Many experiments with particular algorithms have confirmed this. We have already observed that Turing's definition of recursiveness is itself evidence for Church's Thesis. Possibly the best evidence is how well Church's Thesis fits in with the theory. If we replace recursive by computable in results in recursion theory, we often obtain a statement which is evident, or at least more evident than the original result.[6]

We conclude this section with a sample proof of nonexistence of an algorithm. First, we define a real F which is not recursive. We do this by the diagonal method; that is, we make F differ from the recursive function

[6]Kleene *1952b*, Chapter XII gives many further arguments for Church's Thesis. I must admit that in reading that chapter, I am sometimes reminded of Anatole France's version of the Dreyfuss affair in ***Penguin Island***. General Panther fills his office from floor to ceiling with proofs of Dreyfuss' guilt. The Minister of War complains he is spoiling the simplicity of the case. Of course it is good to have proofs, he says, but perhaps it is better to have none at all.

$\{f\}$ at f. Thus we define $F(f) = \{f\}(f) + 1$ if $\{f\}(f)$ is defined and $F(f) = 0$ otherwise. It follows that the set of f such that $\{f\}(f)$ is defined is not recursive; for otherwise, the Enumeration Theorem and VII would show that this F is recursive. Hence if G is the recursive function defined by $G(f) = \{f\}(f)$, then Church's Thesis shows that there is no algorithm by which, given f, we can decide if $G(f)$ is defined.

§3. The arithmetical hierarchy. Kleene left Princeton in 1936 for the University of Wisconsin, where he spent the rest of his career except for visits to other schools and service in the Navy during World War II. One of the first matters he studied there was the effect of putting quantifiers in front of a recursive relation. These results appeared in *1943* and later in *1952b*.

A relation R is *arithmetical* if it has a definition of the form

$$R(\vec{x}) \leftrightarrow Qy_1 \dots Qy_k P(y_1, \dots, y_k, \vec{x})$$

where P is recursive and each Qy_i is either $\exists y_i$ or $\forall y_i$. We call $Qy_1 \dots Qy_k$ the *prefix* and $P(y_1, \dots, y_k, \vec{x})$ the *matrix* of this definition.

We shall show that if in the prefix two adjacent quantifiers are of the same kind (i.e., both $\exists$ or both $\forall$), then they may be replaced by a single such quantifier. Suppose, for example, the definition is $R(x) \leftrightarrow \exists w \exists y \forall z P(w, y, z, x)$. As u varies through all numbers, $(u)_0$, $(u)_1$ varies through all pairs w, y of numbers. Hence we may rewrite our definition as $R(x) \leftrightarrow \exists u\ \forall z\ P((u)_0, (u)_1, z, x)$. Of course, the matrix has changed; but it is still recursive (since $(u)_i$ is a recursive function of u and i).

It follows that every arithmetical relation has a definition in which the prefix does not have two successive quantifiers of the same kind. Then if the prefix has k quantifiers and begins with $\exists$, the relation is Σ^0_k; if the prefix has k quantifiers and begins with $\forall$, the relation is Π^0_k.

If we put a quantifier on x in front of an expression not containing x, we do not change its meaning. From this, we conclude that every recursive relation is Σ^0_k and Π^0_k for all k, and that every Σ^0_k or Π^0_k relation is Σ^0_n and Π^0_n for every $n > k$.

Using the rules for obtaining prenex forms in logic, we obtain closure properties of these relations similar to those given by VI. Thus if P and Q are Σ^0_k, then so are $P \vee Q$ and $P \,\&\, Q$, and the P_1 and P_2 of VI; and $\neg P$ is Π^0_k. If $P_3(\vec{x}) \leftrightarrow \exists y P(y, \vec{x})$ and $P_4(\vec{x}) \leftrightarrow \forall y P(y, \vec{x})$, then P_3 is Σ^0_k and P_4 is Π^0_{k+1}. All of the results also hold with Π and Σ (and $\forall$ and $\exists$) interchanged. These results generally enable us to find the classification of any arithmetical relation we have defined.

The domain of an n-ary function is a class of n-tuples of numbers, i.e., an n-ary relation. We say that a relation is *recursively enumerable* (abbreviated RE) if it is the domain of a recursive function. We say that a relation R is *semicomputable* if there is an algorithm which, applied to $\vec{x}$, gives an output iff $R(\vec{x})$. It is easy to show from Church's Thesis that a relation is RE if it is semicomputable. For this and other reasons, RE relations are of special interest in Recursion Theory.[7]

A relation is Σ^0_1 iff it is RE. For if $P(\vec{x}) \leftrightarrow \exists y R(y, \vec{x})$ with R recursive, then P is the domain of the recursive function $\mu y R(y, \vec{x})$; while if P is the domain of the recursive function with index f, then $P(\vec{x}) \leftrightarrow \exists y T_n(f, \vec{x}, y)$ by the Normal Form Theorem. From this, we can prove that many relations are RE. For example, the set of codes of true formulas is RE; for x is such a code iff there exists a y such that y is the code of a proof and x is the last number in the sequence with code y.

We write W_f for the domain of $\{f\}$, and say that f is an RE-*index* of W_f. Thus every RE relation has at least one RE-index. By the Normal Form Theorem, $W_f(x_1, \ldots, x_n) \leftrightarrow \exists y T_n(f, x_1, \ldots, x_n, y)$. It follows that $W_f(x_1, \ldots, x_n)$ is an RE relation of $f, x_1, \ldots, x_n$. Since this relation enumerates the class of n-ary RE relations, this result is called the RE Enumeration Theorem. It is easy to see that $W_{S(f,y)} = (W_f)_{(y)}$ and similarly for S_m. From this, we may obtain a Recursion Theorem for defining RE relations.

The classification of the arithmetical relations into Σ^0_k and Π^0_k relations is called the arithmetical *hierarchy*. We have yet to see that it is nontrivial, i.e., that Σ^0_k is different from Σ^0_n if $k \neq n$. The key to this is the following important result.

Arithmetical Enumeration Theorem. *For each n and k, there is a Σ^0_k $(n+1)$-ary relation R which enumerates the class of Σ^0_k n-ary relations; and similarly with Π for Σ.*

The proof is by induction on k. The Σ^0_1 case is just the RE Enumeration Theorem. If R is an enumerating relation for the Σ^0_k n-ary relations, then $\neg R$ is an enumerating relation for the Π^0_k n-ary relations. If R is an enumerating relation for the Π^0_k $(n+1)$-ary relations, we may obtain an enumerating relation for the Σ^0_{k+1} n-ary relations by putting an existential quantifier in front of R.

Now let R be the enumerating relation for the Π^0_k unary relations. Define Q by $Q(x) \leftrightarrow \neg R(x, x)$; then Q is Σ^0_k. Since $Q(x) \leftrightarrow \neg R_{(x)}(x)$, $Q \neq R_{(x)}$; so Q is not Π^0_k, and hence not recursive or Σ^0_n or Π^0_n for any $n < k$. Similarly, we can find a set which is Π^0_k, but not Σ^0_k, and hence not recursive or Σ^0_n or Π^0_n for any $n < k$. In particular, we can find an RE set which is not recursive.

A relation R is Δ^0_k if it is both Σ^0_k and Π^0_k. A simple but important result says that a relation is Δ^0_1 iff it is recursive. We already know that recursive relations are Δ^0_1. Suppose that $R(\vec{x}) \leftrightarrow \exists y P(y, \vec{x}) \leftrightarrow \forall y Q(y, \vec{x})$ with P and

[7] RE relations were discovered and studied in the earliest days of recursion theory by Church, Kleene, and Rosser.

Q recursive. Then

$$R(\vec{x}) \leftrightarrow P(\mu y(P(y,\vec{x}) \vee \neg Q(y,\vec{x})),\vec{x});$$

so R is recursive.

To get a similar result for Δ^0_{k+1}, we need the concept of relative recursiveness, which was invented by Turing and has played a large role since. Let Φ be a class of *total* functions. A function is recursive *in* Φ or *relative* to Φ if it belongs to every recursively closed class which includes Φ. Almost all of recursion theory remains valid when *recursive* is replaced by *recursive in* Φ. (In some cases, we must restrict ourselves to finite Φ. This is not a serious difficulty, since it is easy to see that a function is recursive in Φ iff it is recursive in a finite subset of Φ.)

As an example, we can define arithmetical in Φ, Σ^0_k in Φ, etc., and prove all of the above results for these concepts (although the Enumeration Theorem requires that Φ be finite). We also have a new result: a relation is Σ^0_1 in a Π^0_k relation iff it is Σ^0_{k+1}. For if P is Σ^0_{k+1}, then $P(\vec{x}) \leftrightarrow \exists y Q(y,\vec{x})$ where Q is Π^0_k; so P is Σ^0_1 in Q. The converse requires writing down a definition of P and then using the above ideas to show that P is Σ^0_{k+1}.

By applying this result to both R and $\neg R$, we see that R is Δ^0_1 in a Π^0_k relation iff it is Δ^0_{k+1}. From this and the relativization of our result on Δ^0_1, we get:

POST'S THEOREM. *A relation is Δ^0_{k+1} iff it is recursive in a Π^0_k relation.*

When Kleene was about to publish this result, he discovered an abstract by Emil Post containing the same result. He immediately gave it the name Post's Theorem.[8]

There has been much further work on the above ideas, particularly in the field of RE sets. The reader who wants an introduction to the latter subject cannot do better than consult Post's paper [9]; it has been an inspiration to recursion theorists throughout the half century since it was written.

§4. Degrees.

We now come to a topic in recursion theory on which Kleene did little work; he wrote only one joint paper on the subject. However, because of the importance which the subject has achieved and the fundamental character of this paper, it is worth some discussion.

[8]Although careful to not appropriate others' work, Kleene worried little about people appropriating his own. In the fifties, a recursion theorist who should have known better rediscovered the Recursion Theorem (in a slightly different form) and called it the Fixed Point Theorem. In a lecture in which he had used the Recursion Theorem, Kleene merely remarked, "I see some people are now calling this the Fixed Point Theorem."

The concept is simple. If F and G are total[9] functions, let $F \leq G$ mean that F is recursive in G. The relation $\leq$ has some of the properties of a partial ordering; it is reflexive and transitive, but not symmetric. Researchers in partial orderings long ago told us what to do in such a case. We define $F \simeq G$ to mean that $F \leq G \,\&\, G \leq F$. Then $\simeq$ is an equivalence relation. The equivalence class of F is called the *degree* of F, and is designated by $\mathrm{dg}\, F$. We now define $\mathrm{dg}\, F \leq \mathrm{dg}\, G$ to mean that $F \leq G$. The properties of $\leq$ mentioned above guarantee that this definition depends only on $\mathrm{dg}\, F$ and $\mathrm{dg}\, G$, not on F and G, and that the resulting $\leq$ is a partial ordering of the degrees.[10]

Every degree contains a relation; for F is equivalent to the graph of F, which is the set of all $(n+1)$-tuples $(x_1, \ldots, x_n, F(x_1, \ldots, x_n))$. Since the contraction of F is equivalent to F, every degree contains a real and every degree contains a set.

What can we say about the partially ordered set of degrees? The class of recursive total functions is clearly a degree, which is the smallest degree; it is designated by $\mathbf{0}$. Every two element set (and hence every finite set) of degrees have a least upper bound; for the least upper bound of $\mathrm{dg}\, F$ and $\mathrm{dg}\, G$ is $\mathrm{dg}\, H$, where $H(x) = \langle F(x), G(x) \rangle$. Many of the results of the early days of degree theory consisted of showing that this partially ordered set lacks other nice properties.

Post discovered all of this and some deeper results; but he did not publish them. Kleene heard of the results and wrote to Post, saying that he should certainly publish. Post compromised by writing out some things in a very disorganized form and suggesting that Kleene find a graduate student to work out the details. Kleene tried, but the graduate student was not successful; so Kleene (after making his own contribution by proving that two element sets need not have greatest lower bounds) completed the paper and published it as *1954*.

The first important result of *1954* is that the partially ordered set of degrees is not linearly ordered. In terms of functions, this means that there are reals F and G such that neither is recursive in the other. We shall sketch the proof of this, since it is fundamental for all the degree theory which followed.

We want to ensure that $F \neq \{f\}^G$ and $G \neq \{f\}^F$ for all f (where $\{f\}^G$ is the analogue of $\{f\}$ for recursion in G). We shall construct F and G in infinitely many steps, at each of which we define finitely many values of F

[9]It is possible to also consider degrees of partial functions; but there are important differences between the two theories. Most of the research on degrees have been on degrees of total functions.

[10]In proving results about degrees, equivalence classes are difficult to deal with; so we usually pick one function in the equivalence class and deal with it. Then why bother with the equivalence classes at all? Because the theory of partially ordered sets is well-developed and suggests many problems about degrees, which have been the heart of the theory.

and G. At the end of the fth step, we will have insured that $F \neq \{f\}^G$ and $G \neq \{f\}^F$, and that $F(f)$ and $G(f)$ are defined. At the fth step, pick a number x such that $F(x)$ is not yet defined. If there is no way to complete the definition of G so that $\{f\}^G(x)$ is defined, then we ensure $F \neq \{f\}^G$ by setting $F(x) = 0$. Suppose we can complete the definition of G so that $\{f\}^G(x) = y$. It is easy to see from the definition of *recursive in G* that the truth of the equation $\{f\}^G(x) = y$ depends only on the values of G at finitely many arguments. Hence at step f, we may insure that $\{f\}^G(x) = y$; and we may then insure that $F \neq \{f\}^G$ by defining $F(x) = y + 1$. By defining further values of F and G, we may also insure that $G \neq \{f\}^F$ and that $F(f)$ and $G(f)$ are defined.

Before publishing this paper, Kleene showed it to his graduate student Clifford Spector, who solved some of the problems which Kleene and Post left open. In particular, he proved in [12] that there were minimal degrees, i.e., degrees $\boldsymbol{a} > \mathbf{0}$ such that there is no degree $\boldsymbol{b}$ with $\mathbf{0} < \boldsymbol{b} < \boldsymbol{a}$. However, neither Kleene nor Spector did any further work in degrees.

We cannot even summarize all the work in degrees that has taken place since then; so we will only mention the most important single development. Call a degree RE if it contains a RE relation. The degree $\mathbf{0}'$ of a binary RE relation which enumerates all RE sets is easily seen to be the largest RE degree. Since there is an RE relation which is not recursive, $\mathbf{0}' \neq \mathbf{0}$; so there are at least two RE degrees. Post had long been interested in the problem of whether there were additional RE degrees. It would certainly suffice to show that there are incomparable RE degrees; but the Kleene-Post proof will not do, since the F and G constructed there are not RE. For this, we should need the construction to be effective; but there is no effective way of deciding if we can complete the definition of G in the way required.

The additional idea needed to make the construction effective is the Priority Method, which was discovered independently by the undergraduate Friedberg[11] in the US and the undergraduate Muchnik in Russia not long after the publication of the Kleene-Post paper. (See [10] for references and for much else about priority.) In the forty years since then, this method has been intensively studied, but we are still very far from discovering all of its possibilities.

§5. Real arguments and the analytical hierarchy. Suppose we take the functions we have been considering up to now and use them as arguments or values for new functions. How shall we extend recursion theory to this

[11] Friedberg, who was at Harvard, learned of Post's problem in a course given by Hartley Rogers at MIT. This course seems to have attracted some of the best young minds in Boston. David Mumford, who has since made his name well-known in other fields, is still known to some elder logicians as the second best student in Rogers' recursion theory course.

situation? Coding is of no use here; for the set of functions is uncountable and hence cannot be coded by numbers. Kleene gave much thought to this question from the time he returned from the war. When he had a satisfactory theory, and he and his graduate students had solved some of the resulting problems, he published his results in a series of three important papers (*1955a*, *1955b*, *1955c*).

First, we have to decide which functions to allow as arguments and values. There is no need to consider functions as values; for if F takes functions as values, it may be replaced by the F' defined by $F'(x, y) = (F(x))(y)$. We will allow only total functions as arguments. By replacing a total function by its contraction, we may assume that the arguments are reals.

We shall thus henceforth consider functions which take reals and numbers as arguments and numbers as values. We shall always put the real arguments first. A function is (m, n)-ary if it has m real arguments and n number arguments. We use α, β and γ as variables for reals.

How can we extend computability to such functions? The problem here is: how are we to be given a real argument α of our function? The obvious answer is that we are given an object which, when presented with any given number x, will give us $\alpha(x)$. Following Turing, we call such an object an *oracle* for α.

Let F be an (m, n)-ary function. An *associate* of F is a total function G of number variables such that if $G(\overline{\alpha}_1(z), \ldots, \overline{\alpha}_m(z), x_1, \ldots, x_n) = y+1$, then $F(\alpha_1, \ldots, \alpha_m, x_1, \ldots, x_n) = y$, and such that if $F(\alpha_1, \ldots, \alpha_m, x_1, \ldots, x_n) = y$, then $G(\overline{\alpha}_1(z), \ldots, \overline{\alpha}_m(z), x_1, \ldots, x_n) = y + 1$ for all sufficiently large z. It is easy to see that F is computable iff it has a computable associate. We therefore define an (m, n)-ary function to be recursive if it has a recursive associate.[12]

Extending the results of §2 to our new functions is mostly an exercise in constructing associates. There are also a few new results; e.g., $\alpha(x)$ and $\overline{\alpha}(x)$ are recursive functions of α and x. (On the other hand, $\alpha = \beta$ is not a recursive function of α and β.) As an index of a recursive F, we use any index of an associate of F. For $m \neq 0$, the Normal Form Theorem now has the form

$$\{f\}(\alpha_1, \ldots, \alpha_m, x_1, \ldots, x_n) = U(\mu y T_{m,n}(f, \overline{\alpha}_1(y), \ldots, \overline{\alpha}_m(y), x_1, \ldots, x_n)).$$

We now let $F_{(y)}$ be the function obtained from F by substituting y for the first number argument to F. We then have the Recursion Theorem as before.

In III, we substituted a recursive function for a number argument of a recursive function G. We now do the same for real arguments. Of course, we cannot substitute $H(\ldots)$ for α, since the former is a number and the latter

[12]There are many satisfactory definitions of recursive in this case. The one that we use here comes from Kleene *1959b*.

is a real; so we need some new notation. If A is an expression containing x which designates a number, then $\lambda x A$ designates the function whose value at any x is A. (This is the λ which gives the name to λ-definable functions.) Clearly $\lambda x A$ is a unary function, which is a real iff A is defined for every value of x. The proof of the following consists of a straightforward construction of an associate for F.

SUBSTITUTION THEOREM. *If G and H are recursive and H is total, then the F defined by $F(\vec{\alpha}, \vec{x}) = G(\lambda z H(\vec{\alpha}, \vec{x}, z), \vec{\alpha}, \vec{x})$ is recursive.*

What about relativization to a class Φ of total functions? If the functions in Φ have real arguments, there are difficulties; this will be considered further in §7. If Φ is a class of functions of number variables, we may define relative recursiveness as before. However, there is in this case another simple definition: F is recursive in Φ iff there are $G_1, \dots, G_j \in \Phi$ and a recursive F' such that $F(\vec{\alpha}, \vec{x}) = F'(\langle G_1 \rangle, \dots, \langle G_j \rangle, \vec{\alpha}, \vec{x})$ for all $\vec{\alpha}$ and $\vec{x}$.

Now let us turn to the generalization of the arithmetical hierarchy. Here, of course, we wish to allow both number quantifiers and real quantifiers. An *analytical* relation is one which has a definition consisting of number and real quantifiers followed by a recursive relation.

By a proof only slightly more complicated than the one we used in the arithmetical hierarchy, we show that every analytical relation has a definition in which the prefix does not have two successive quantifiers of the same kind, and there is exactly one number quantifier, which comes last. Then if the prefix has k real quantifiers and beings with $\exists$, the relation is Σ^1_k; if the prefix has k real quantifiers and begins with $\forall$, the relation is Π^1_k.

We can say a bit more about the matrix. Suppose that R is a $(0, 1)$-ary Π^1_1 relation, so that $R(x) \leftrightarrow \forall\alpha\exists y P(\alpha, y, x)$ with P recursive. Let $F(\alpha, x) = \mu y P(\alpha, y, x)$ and let f be an index of F. Then by the Normal Form Theorem,

$$R(x) \leftrightarrow \forall\alpha(F(\alpha, x) \text{ is defined}) \leftrightarrow \forall\alpha\exists y T_{1,1}(f, \overline{\alpha}(y), x).$$

The general conclusion is: the definition may be chosen so that if α and y and the last two variables in the prefix, then they occur in the matrix only in the context $\overline{\alpha}(y)$.

As before, we can show that every arithmetical relation is Π^1_k and Σ^1_k for all k, and that every Σ^1_k or Π^1_k relation is Σ^1_n and Π^1_n for all $n > k$. We can also find closure properties similar to the arithmetical case. In particular, prefixing number quantifiers does not change the position of a relation in the analytical hierarchy. We can prove an Analytical Enumeration Theorem (again using the RE Enumeration Theorem) and use it to show that the hierarchy is nontrivial.

We shall establish a connection between Π^1_1 sets and ordinals. We first define a set O_σ for each ordinal σ by induction on σ as follows: $e \in O_\sigma$ if $W_e \subset \bigcup_{\tau<\sigma} O_\tau$. We let O be[13] the union of the O_σ. If $e \in O$, $|e|$ is the least σ such that $e \in O_\sigma$. An ordinal σ is *recursive* if $\sigma = |e|$ for some $e \in O$; any such e is then called a *notation* for σ.

Since O_σ is a nondecreasing function of σ, we have

$$(1) \qquad e \in O_\sigma \leftrightarrow e \in O \,\&\, |e| \leq \sigma.$$

Hence, setting $O^*_\sigma = \bigcup_{\tau<\sigma} O_\tau$,

$$(2) \qquad e \in O^*_\sigma \leftrightarrow e \in O \,\&\, |e| < \sigma.$$

Lemma. For every e, $e \in O$ iff $W_e \subset O$; in this case, $|e|$ is the least ordinal greater than $|f|$ for all $f \in W_e$.

If $e \in O$, then $e \in O_{|e|}$; so $W_e \subset O^*_{|e|}$; so $W_e \subset O$ and $|f| < |e|$ for $f \in W_e$ by (2). If $\sigma < |e|$, $e \notin O_\sigma$; so there is an $f \in W_e$ such that $f \notin O^*_\sigma$ and hence $\sigma \leq |f|$ by (2). If $W_e \subset O$, then $W_e \subset O^*_\sigma$ for some σ; so $e \in O_\sigma \subset O$.

From the Lemma, it is easy to show that if σ is recursive, then $\sigma + 1$ is recursive. It follows that every finite ordinal is recursive. A little more work shows that most of the countable ordinals which occur in practice (e.g., ω and ε_0) are recursive.

A sequence $\{x_n\}$ is *O-descending* if $x_{n+1} \in W_{x_n}$ for all n. We show that $x \in O$ iff there is no O-descending sequence beginning with x. If $x \in O$ and $\{x_n\}$ is an O-descending sequence with $x_0 = x$, then $x_{n+1} \in O$ and $|x_{n+1}| < |x_n|$ for all n by induction on n and the Lemma; and this is impossible. If $x \notin O$, the Lemma shows that we can choose inductively a O-descending sequence $\{x_n\}$ such that $x_0 = x$ and $x_n \notin O$ for all n. Thus we have

$$x \in O \leftrightarrow \neg\exists\alpha(\alpha(0) = x \,\&\, \forall n(\alpha(Sc(n)) \in W_{\alpha(n)}));$$

so O is Π^1_1.

We shall show that O is, in a certain sense, the most complicated Π^1_1 set. We say that a set A is *many-one reducible* to a set B if there is a recursive real F such that $x \in A \leftrightarrow F(x) \in B$ for all x. This clearly implies that A is recursive in B.

Theorem. *Every Π^1_1 set is many-one reducible to O.*

If A is Π^1_1, we have $x \in A \leftrightarrow \forall\alpha\exists y R(x, \overline{\alpha}(y))$ where R is recursive. Let $z \in O$. Let $w\hat{\ }m$ be a recursive function of w and m such that if $w =$

[13]Our O is a simplification of the O defined by Church and Kleene and used throughout Kleene's work. (Caution: Our $<_O$ is somewhat different from the $<_O$ of Church and Kleene.)

$\langle x_1, \ldots, x_n\rangle$, then $w\hat{\ }m = \langle x_1, \ldots, x_n, m\rangle$. Then $\exists m(\{f\}(x, w\hat{\ }m) = y)$ is an RE relation of x, f, w, y. Let q be an RE-index of this relation. Using the Recursion Theorem, define a recursive total F with index f by

$$\begin{aligned} F(x,w) &= z && \text{if } R(x,w),\\ &= S_3(q,x,f,w) && \text{otherwise.} \end{aligned}$$

We complete the proof by showing that $x \in A \leftrightarrow F(x, \langle\rangle) \in O$. Suppose that $x \notin A$, and choose α so that $\forall y \neg R(x, \overline{\alpha}(y))$. Then $F(x, \overline{\alpha}(y))$ is an RE-index of the set of $F(x, \overline{\alpha}(y)\hat{\ }m)$, which contains $F(x, \overline{\alpha}(y+1))$. Hence $\{F(x, \overline{\alpha}(n))\}$ is an O-descending sequence; so $F(x, \langle\rangle) \notin O$. Now suppose that $F(x, \langle\rangle) \notin O$. We will define $\alpha(n)$ by induction so that for all n, $F(x, \overline{\alpha}(n)) \notin O$. Then by the definition of F, $\forall n \neg R(x, \overline{\alpha}(n))$; so $x \notin A$. We have $F(x, \overline{\alpha}(0)) \notin O$. Suppose $\overline{\alpha}(n)$ has been defined and $F(x, \overline{\alpha}(n)) \notin O$. Then $F(x, \overline{\alpha}(n)) = S_3(q, x, f, w)$. By the Lemma, there is an m such that $F(x, \overline{\alpha}(n)\hat{\ }\langle m\rangle) \notin O$. Letting $\alpha(n) = m$, we have $F(x, \overline{\alpha}(n+1)) \notin O$.

Corollary. O is not Σ^1_1.

Every set many-one reducible to a Σ^1_1 set is Σ^1_1. Hence if O were Σ^1_1, then every Π^1_1 set would be Σ^1_1, which is not the case.

A relation is Δ^1_n if it is both Π^1_n and Σ^1_n. Recall that a relation is Δ^0_1 iff it is recursive. We shall give a similar (but much more difficult) characterization of Δ^1_1. This can be done for all relations; but, following Kleene, we shall consider only relations of number variables. In fact, we consider only sets, since n-ary relations can be replaced by their contractions.

We define the set H_σ by induction on σ as follows: $H_\sigma(x) \leftrightarrow (\exists\tau < \sigma)\neg W_x(H_\tau)$ (where W_x is the $(1,0)$-ary relation with RE-index x). A set is *hyperarithmetical*[14] if it is many-one reducible to H_σ for some recursive σ. We shall show that a set is Δ^1_1 iff it is hyperarithmetical.

We first obtain an inductive definition of H_σ which uses notations instead of ordinals. Let $f <_O e$ mean that there is a sequence $z_1, \ldots, z_n$ with $n > 1$, $z_1 = f$, $z_n = e$, and $z_i \in W_{z_{i+1}}$ for $1 \le i < n$. Writing this out (using a code of the sequence), we see that the relation $<_O$ is Σ^0_1. If $e \in O \,\&\, f <_O e$, then $f \in O \,\&\, |f| < |e|$ by the Lemma. If $e \in O \,\&\, \sigma < |e|$ then there is an $f <_O e$ such that $|f| = \sigma$; this is easily proved by induction on $|e|$, using the Lemma. It follows that for $e \in O$,

$$H_{|e|}(x) \leftrightarrow \exists f(f <_O e \,\&\, \neg W_x(H_{|f|})). \tag{3}$$

[14]The hyperarithmetical sets were discovered independently by Kleene, Martin Davis, and Mostowski. However, it was Kleene who discovered the relationship of hyperarithmetical relations to the analytical hierarchy. I discovered the definition given here about 1962.

We write $f \leq_O e$ for $f <_O e \vee f = e$. For $e \in O$, define $P_e(f, x) \leftrightarrow f \leq_O e \,\&\, H_{|f|}(x)$. Fix $e \in O$. Suppose that $P = P_e$. Then

$$\forall f \forall x (P(f, x) \rightarrow f \leq_O e). \tag{4}$$

If $f \leq_O e$, $P_{(f)} = H_{|f|}$. If also $g <_O f$, then $g \leq_O e$, so $P_{(g)} = H_{|g|}$. Thus by (3),

$$\forall f \forall x (f \leq_O e \rightarrow (P(f, x) \leftrightarrow \exists g (g <_O f \,\&\, \neg W_x(P_{(g)})))). \tag{5}$$

Conversely, suppose that (4) and (5) hold. Using induction on $|f|$ and (5), we get $P_{(f)} = H_{|f|}$ for $f \leq_O e$. From this and (4), $P = P_e$.

Let $Q(\alpha, e)$ be the conjunction of $\forall f \forall x (\alpha(\langle f, x \rangle) = 0 \rightarrow f \leq_O e)$ and

$$\forall f \forall x (f \leq_O e \rightarrow (\alpha(\langle f, x \rangle) = 0 \leftrightarrow \exists g (g <_O f \,\&\, \neg W_x(\lambda x \alpha(\langle g, x \rangle))))).$$

By the RE Enumeration Theorem and the Substitution Theorem, Q is arithmetical. The above shows that for $e \in O$, $Q(\alpha, e)$ iff $\forall f \forall x (\alpha(\langle f, x \rangle) = 0 \leftrightarrow P_e(f, x))$.

We can now prove that every hyperarithmetical set is Δ^1_1. It suffices to show that H_σ is Δ^1_1 for all recursive σ. Let e be a notation for σ. Then

$$H_\sigma(x) \leftrightarrow \forall \alpha (Q(\alpha, e) \rightarrow \alpha(\langle e, x \rangle) = 0) \leftrightarrow \exists \alpha (Q(\alpha, e) \,\&\, \alpha(\langle e, x \rangle) = 0).$$

Thus H_σ is Δ^1_1. In fact, we have proved that there is a Π^1_1 relation H_Π and a Σ^1_1 relation H_Σ such that $H_{|e|}(x) \leftrightarrow H_\Pi(e, x) \leftrightarrow H_\Sigma(e, x)$ for $e \in O$.

We next show that there is a recursive real J such that $x \in O^*_\sigma \leftrightarrow J(x) \in H_\sigma$ for all σ. We assume that J is defined and prove the equivalence by induction on σ. It is enough to prove that $x \in O_\tau \leftrightarrow \neg W_{J(x)}(H_\tau)$ for $\tau < \sigma$. Now $x \in O_\tau \leftrightarrow (\forall y \in W_x)(y \in O^*_\tau)$; so by the induction hypothesis, $x \in O_\tau \leftrightarrow (\forall y \in W_x)(J(y) \in H_\tau)$. Thus we have to show that $W_{J(x)}(H_\tau) \leftrightarrow (\exists y \in W_x)(J(y) \notin H_\tau)$. By the RE Enumeration Theorem, the right side is an RE relation of H_τ, x, j where j is an index of J. If q is an RE-index of this relation, we want to have $J(x) = S_2(q, x, j)$. This can be achieved by the Recursion Theorem.

It follows that for σ recursive, O^*_σ is hyperarithmetical. Now if σ is recursive, so is $\sigma+1$, and $O_\sigma = O^*_{\sigma+1}$ by (1) and (2). Hence O_σ is hyperarithmetical.

Boundedness Theorem. *If A is a Σ^1_1 subset of O, then $A \subset O_\sigma$ for some recursive σ.*

If not, it follows from (1) that every recursive σ is $< |x|$ for some $x \in A$. It follows by (2) that

$$\begin{aligned} e \in O &\leftrightarrow (\exists x \in A)(e \in O^*_{|x|}) \\ &\leftrightarrow (\exists x \in A)(J(e) \in H_{|x|}) \\ &\leftrightarrow \exists x (x \in A \,\&\, H_\Sigma(J(e), x)). \end{aligned}$$

Hence O is Σ^1_1, contradicting the Corollary above.

Now we shall prove that every Δ^1_1 set is hyperarithmetical. Let A be Δ^1_1. Since A is Π^1_1, there is a recursive real F such that $x \in A \leftrightarrow F(x) \in O$. If $B = \{F(x) \ : \ x \in A\}$, then $y \in B \leftrightarrow \exists x(x \in A \,\&\, y = F(x))$; so B is Σ^1_1. By the Boundedness Theorem, there is a recursive σ such that $B \subset O_\sigma$. It follows that $x \in A \leftrightarrow F(x) \in O_\sigma$; so A is many-one reducible to O_σ. Since O_σ is hyperarithmetical, A is hyperarithmetical. (This proof, which is different from Kleene's original proof, comes from the thesis of his student Spector [11].)

One might hope to extend this result in the same way that the characterization of Δ^0_1 was extended by Post's Theorem. The desired result would say that a set is Δ^1_{n+1} iff it is hyperarithmetical in a Π^1_n relation. However, this is false, as Addison and Kleene showed in *1957*.

More information about the H_σ is given by the next result.

Theorem. *Let ρ be a recursive ordinal, and let A be Π^0_1 in H_ρ. Then there is a recursive real F such that $x \in A \leftrightarrow F(x) \in H_\sigma$ for all $\sigma > \rho$.*

Let e be a notation for ρ. If J is as above, then $\rho < \tau \leftrightarrow H_\tau(J(e))$ for all τ by (2). Hence $\rho = \tau \leftrightarrow H_{\tau+1}(J(e)) \,\&\, \neg H_\tau(J(e))$. From this and the definition of H_τ, $\rho = \tau \leftrightarrow \neg W_{J(e)}(H_\tau) \,\&\, \neg H_\tau(J(e))$.

There is an RE Q such that $x \in A \leftrightarrow \neg Q(H_\rho, x)$. Then for $\sigma > \rho$,

$$\begin{aligned} x \in A &\leftrightarrow (\exists \tau < \sigma)(\rho = \tau \,\&\, \neg Q(H_\tau, x)) \\ &\leftrightarrow (\exists \tau < \sigma)\neg(W_{J(e)}(H_\tau) \vee H_\tau(J(e)) \vee Q(H_\tau, x)). \end{aligned}$$

Hence if q is an RE-index of the relation $W_{J(e)}(\alpha) \vee \alpha(J(e)) = 0 \vee Q(\alpha, x)$ of α, x and $F(x) = S(q, x)$, then $x \in A \leftrightarrow F(x) \in H_\sigma$.

In particular, it follows that if $\rho < \sigma$, then H_ρ is many-one reducible to H_σ. On the other hand, H_σ is not recursive in H_ρ; for if it were, then every set Π^0_1 in H_ρ would be recursive in H_ρ, which is false. Using the theorem and Post's Theorem, it is not difficult to show that a set is Π^0_n iff it is many-one reducible to H_n. It follows that H_ω is an example of a hyperarithmetical relation which is not arithmetical.

There has been considerably more study of the hyperarithmetical sets. In particular, Kleene *1959c* showed that hyperarithmetical sets could be defined in other interesting ways.

In his thesis [1], Kleene's student Addison studied the analytic hierarchy relative to the set Φ of all reals, or, as it is usually called, the *projective* hierarchy. There is a natural topology on the set of all $\alpha_1, \dots , \alpha_m, x_1, \dots , x_n$ in which the open sets are the Σ^0_1 in Φ sets. Then Π^1_k *in* Φ and Σ^1_k *in* Φ can be given topological definitions. These concepts turn out to be identical with concepts studied in the 1920's by Lusin, Suslin, Sierpiński, and others. For example, the theorem that Δ^1_1 in Φ is the same as hyperarithmetical in Φ is a famous theorem of Suslin; so the equivalence of Δ_1^1 and hyperarithmetical is usually called the Suslin-Kleene Theorem.

§**6. Realizability.** As Yiannis Moschovakis observed in a brief talk on Kleene's work at an ASL meeting, Kleene rarely worked on problems formulated by others. In particular, he did little on the application of recursion theory to other fields. By far his most important work on applications was in the field of intuitionism, where the problems to be solved were not clear before he entered the field. We shall give only a brief summary of the large amount of work he did in this field between 1940 and 1969, since there is an excellent survey of this work by Kleene *1973*.

Kleene became interested in intuitionism by reading the works of Brouwer, Heyting, Gödel, Gentzen and others on the subject. His interest was a practical one; he felt that recursive functions must have a role in explaining the nature of a constructive proof, which is what intuitionism is all about. This consideration led him to the idea of realizability, which he developed in various forms over the years. The conclusions did not fulfill Kleene's greatest hope of giving a complete explanation of intuitionism in terms of recursive functions; but they did give many interesting results and much understanding.

Since Brouwer was quite prejudiced against formal systems, the formalization of intuitionism was left to his successors. By 1940, the formalization of the intuitionistic predicate calculus had been agreed upon. The notation is the same as the classical predicate calculus; but intuitionistically, none of the logical symbols $\neg$, $\vee$, $\&$, $\rightarrow$, $\forall$ and $\exists$ can be defined in terms of the others. The intuitionistically valid sentences are a proper subset of the classically valid sentences. An example of a classically valid but not intuitionistically valid sentence is $A \vee \neg A$; this is intuitionistically true only if we have a constructive method of deciding which of A and $\neg A$ is true.

Intuitionistic arithmetic is formed from the intuitionistic predicate calculus by adding the symbols 0 and Sc, the Peano axioms, and symbols for some basic functions together with the defining equations for these functions. (The exact choice of these functions is not too important. We shall suppose that they include the T_n.) In arithmetic, we can eliminate $\neg$ by defining $\neg A$ to mean $A \rightarrow 0 = 1$.

One explanation of intuitionism is this: to say a mathematical statement A is true is to say that one has a construction which demonstrates A. Kleene's idea was to take *e realizes A* to mean that the number e codes a construction which demonstrates A. After some false starts, he arrived in *1945* at the following definition (by induction on the length of the sentence A of arithmetic):

(a) If A is prime (or, as it is usually called nowadays, atomic), e realizes A if A is true;
(b) If A is $B \,\&\, C$, e realizes A if $(e)_0$ realizes B and $(e)_1$ realizes C;
(c) If A is $B \vee C$, e realizes A if either $(e)_0 = 0$ and $(e)_1$ realizes B or $(e)_0 \neq 0$ and $(e)_1$ realizes C;
(d) If A is $B \rightarrow C$, e realizes A if for each a, if a realizes B, then $\{e\}(a)$ is defined and realizes C;
(e) If A is $\forall x B(x)$, e realizes A if for each k, $\{e\}(k)$ is defined and realizes $B(k)$.
(f) If A is $\exists x B(x)$, e realizes A if $(e)_0$ realizes $B((e)_1)$.

We then say that A is *realizable* if some e realizes A. Note that $\neg A$ is realizable iff A is not realizable.

Kleene conjectured that if A was a theorem of intuitionistic arithmetic, then A was realizable. This conjecture was verified by Kleene's first Ph.D. student, David Nelson, in his thesis [6].

One consequence of this result was that certain sentences which could be proved (nonconstructively) in classical arithmetic were not provable in intuitionistic arithmetic. For example, let $A(x)$ be the formula $\exists z T_1(x, x, z)$. Then $\forall x(A(x) \vee \neg A(x))$ is provable in the classical predicate calculus. Suppose that it is provable in intuitionistic number theory and hence realized by some e. Then for each x, $\{e\}(x)$ is defined and realizes $A(x) \vee \neg A(x)$. It follows that $A(x)$ is realizable iff $(\{e\}(x))_0 = 0$. But $A(x)$ is realizable iff $T_1(x, x, z)$ is realizable for some z; and $T_1(x, x, z)$ is realizable iff it is true. Thus $\exists z T_1(x, x, z)$ iff $(\{e\}(x))_0 = 0$, and hence $\exists z T_1(x, x, z)$ is a recursive relation. Since $\exists z T_1(x, x, z)$ iff $\{x\}(x)$ is defined, this contradicts the result at the end of §2. A consequence of this is that the classically false sentence $\neg\forall x(A(x) \vee \neg A(x))$ is realizable.

Kleene saw that the usefulness of Nelson's result could be increased by modifying the notation of realizability by replacing *e realizes A* at some places by *e realizes A and A is a theorem*. (Specifically, replace $(e)_1$ *realizes B* and $(e)_1$ *realizes C* in clause (c), *a realizes B* in clause (d), and $(e)_0$ *realizes* $B((e)_1)$ in clause (f).) Nelson's result still goes through, and it now has the following easy corollaries:

(i) If $A \vee B$ is a theorem, then either A or B is a theorem.
(ii) If $\exists x A(x)$ is a theorem, there is a number k such that $A(k)$ is a theorem.
(iii) If $\forall x \exists y A(x, y)$ is a theorem, then there is a recursive real F such that $A(k, F(k))$ is a theorem for each number k.

The last of these results confirms a conjecture that Kleene made at the beginning of his consideration of intuitionism. Kleene later (in *1962a*) noticed that (i) and (ii) could be proved by a much simpler definition of *realizes* which did not involve recursive functions at all.

Beginning about 1950, Kleene considered the same problems for second order number theory, or, as it is often called, analysis. This work, together with related work of his student Vesley, is contained in *1965*.

To obtain intuitionistic analysis from intuitionistic arithmetic, we add real variables. (It is not necessary to add variables for n-ary functions, since the coding of finite sequences can be carried out in intuitionistic arithmetic.) This, of course, requires extending the intuitionistic predicate calculus to the new variables; but this is no problem. As further axioms, we first need for each numerical term $t(x)$ an axiom saying there is one and only one real α such that $\alpha(x) = t(x)$ for all x. There is a weak form of the axiom of choice expressible in this notation which is intuitionistically valid. We take as an axiom Brouwer's bar theorem, which says that certain types of transfinite induction are allowed.

The last axiom, which Kleene calls Brouwer's principle, is a strong form of the statement: if to each α a number $y(\alpha)$ is assigned, then for each α the value of $y(\alpha)$ depends only on a finite number of values of α. This is intuitionistically valid because $y(\alpha)$ must be obtained constructively from an oracle for α; so we can only consult the oracle finitely often. Unlike the other axioms, it is not classically valid, as we see by taking $y(\alpha) = 0$ if $\forall x(\alpha(x) = 0)$ and $y(\alpha) = 1$ otherwise.

The extension of realizability from arithmetic to analysis presents considerable difficulty. In arithmetic, we consider only sentences; formulas with free variables were treated by replacing each free variable by a name of a number. This does not work in analysis, since we do not have names for all reals. Thus we must now define realizability for a pair (A, Φ), where Φ is a function which assigns a meaning to each variable free in A. Now the meaning of a real variable may well be a nonrecursive real; so we can expect the construction for (A, Φ) to have some nonconstructive elements. This is best handled by allowing (A, Φ) to be realized by a real instead of a number.

We are thus led to define ε Φ-*realizes* A for ε a real and A a formula of analysis. The definition is virtually a copy of the definition for arithmetic. We take $(\varepsilon)_i$ to be the real defined by $(\varepsilon)_i(x) = (\varepsilon(x))_i$. The explanation of $\{\varepsilon\}(\alpha)$ (which occurs in the definition of ε *realizes* $A \to B$ and ε *realizes* $\forall\alpha A(\alpha)$) is that $\{\varepsilon\}(\alpha) = F(\alpha)$ where ε is an associate of F.[15]

With this definition, it is fairly straightforward to show that every theorem of intuitionistic analysis is realizable. One consequence is that intuitionistic analysis is seen classically to be consistent; this is not trivial, since intuitionistic analysis is not a subsystem of classical analysis. As in arithmetic, realizability could be used to show that certain classically provable results

[15] If A is a sentence of arithmetic, realizability in the new sense is not always the same as realizability in the old sense. For example, the classically true but unrealizable sentence discussed above is now realized by a suitable nonrecursive ε.

were not intuitionistically provable. Other interesting independence results could be obtained by restricting in some way the class of reals allowed to realize a formula. For example, one can show that the bar theorem is not provable from the remaining axioms. (This has some interest, since Brouwer gave various proofs of the bar theorem from results he thought to be simpler.)

Extending (i)–(iii) to analysis proved to be quite difficult. Kleene's student Joan Rand Moschovakis [5] was able to extend (i) and (ii), but only at the cost of replacing Brouwer's principle by a weaker axiom. Kleene finally found a proof of (i)–(iii) in *1969*. It required carrying out the definition of realizability within the language of analysis. This completed Kleene's work on realizability; but it has since been used by several investigators, usually to prove independence results.

§7. Objects of higher types. In §5, we added some of the functions of our theory as new arguments, thus obtaining new functions. This process can be repeated once more, or even infinitely many times. Kleene began thinking about the resulting theory about 1952. The principal results appeared in a two part paper (*1959a*, *1963*) which contains some of Kleene's most profound work.

We first define the *objects of type t* by induction on t. An object of type 0 is a number; an object of type $t+1$ is a mapping of the set of objects of type t into the set of numbers. Thus an object of type 1 is just a real. We use α^t, β^t, and γ^t as variables for objects of type t; we omit the superscript whenever possible.

The functions we now consider will have n arguments, each of a specified type, and will have numbers as values. In describing computability of such functions, a new problem arises. Again we are given an argument α^{t+1} by being given an oracle which, presented with a β^t, will give us the value $\alpha(\beta)$. The trouble is that the object β may be highly noncomputable; so we will have introduced noncomputable information into our computation.

One solution is to require the oracle to give the value $\alpha(\beta)$ after having been presented with only finitely many values of β. This means that the α must be of a special kind; its value at any β is determined by only finitely many values of β. The recursion theory of these objects was developed by Kreisel and Davis (independently), and Kleene eventually took up the question in *1959b*. As Kleene showed, this theory reduces to the theory in §5. (For references and more information about this recursion theory, see [7].)

Kleene was, however, interested in a theory which would really use the objects of higher type. His idea was that an oracle for α would give the value of $\alpha(\beta)$ if it were presented with an algorithms for computing β. (The algorithm could use any oracles which we already had on hand, including an oracle for α.) This idea could be mixed with any of the definitions of recursive we have mentioned (although in each case there are some complications). Kleene obtained his first results by using λ-definability; but then he saw how to generalize recursively closed sets. He eventually published papers (*1962b*, *1962c*, *1962d*, *1962e*) showing that all these approaches were equivalent.

We are going to modify I–V to fit the present situation. We add to I three functions Pr, DC, and Ap defined by

$$Pr(0) = 0, Pr(x+1) = x,$$

$$DC(0, y, z) = y, DC(x+1, y, z) = z,$$

$$Ap(\alpha^1, x) = \alpha(x).$$

II and III remain the same, except that the variables may be of any types. For reasons we shall explain shortly, we omit the old IV and V. Our new IV corresponds to asking the oracle for an object of type ≥ 2 for a value of the object. (Oracles for reals are taken care of by the function Ap.)

IV. If G is recursive and F is defined by $F(\alpha^{t+2}, \vec{\beta}) = \alpha(\lambda\delta^t G(\alpha, \vec{\beta}, \delta))$, then F is recursive.

We assign indices to all these functions much as before. In particular, if F and G are as in IV and g is an index of G, then $\langle 4, g \rangle$ is an index of F. We again want to have the Enumeration Theorem: $\{f\}(\vec{\alpha})$ is a recursive function of $f, \vec{\alpha}$. Our new V says that this function is recursive for each choice of the types of the α_i.

V. If F is defined by $F(f, \vec{\alpha}) = \{f\}(\vec{\alpha})$, then F is recursive.

To the F of V we assign the index $\langle 5, \langle t_1, \dots, t_n \rangle \rangle$, where t_i is the type of α_i. Note that in V, F is defined in terms of all recursive functions together with their indices. Hence it is impossible to define the recursive functions by I–V as before. Instead, we will define the true formulas to be the theorems of a formal system, and then use this to define the recursive functions and their indices.

A formula is now an equation $\{f\}(\vec{\alpha}) = y$, where f is an index of a function and the types of α_i are such that $\vec{\alpha}$ is an appropriate argument for $\{f\}$. The formal system is very much like the formal system in §2; so we shall give only the rule of inference corresponding to IV. It says that $\{\langle 4, g \rangle\}(\alpha^{t+2}, \vec{\beta}) = \alpha(\gamma)$ is inferable from the set of formulas $\{g\}(\alpha, \vec{\beta}, \delta) = \gamma(\delta)$ for δ an object of type t. Since this rule has infinitely many hypotheses, we can no longer define proofs as before. Instead, we define a class T_σ of formulas for each ordinal σ by induction on σ as follows: a formula is in T_σ if it is inferable from formulas in $\bigcup_{\tau<\sigma} T_\tau$. Note that T_σ is a non-decreasing function of σ. A formula is a *theorem* if it belongs to T_σ for some σ.

A class T of formulas is *regular* if for each f and $\vec{\alpha}$, there is at most one y such that $\{f\}(\vec{\alpha}) = y$ is in T. We show that the class of theorems is regular. It will clearly suffice to prove that for each σ, T_σ is regular; and we do this by induction on σ. We suppose $f = \langle 4, g \rangle$; the proof is much the same for other values of f. Suppose that $\{f\}(\alpha, \vec{\beta}) = y$ and $\{f\}(\alpha, \vec{\beta}) = y'$ are in T_σ. Then there are γ and γ' such that $y = \alpha(\gamma)$, $y' = \alpha(\gamma')$, and for each δ, $\{g\}(\alpha, \vec{\beta}, \delta) = \gamma(\delta)$ and $\{g\}(\alpha, \vec{\beta}, \delta) = \gamma'(\delta)$ are in $\bigcup_{\tau<\sigma} T_\tau$. By the induction hypothesis, $\gamma(\delta) = \gamma'(\delta)$ for all δ; so $\gamma = \gamma'$; so $y = y'$.

We let $\{f\}(\vec{\alpha})$ be the unique number y such that $\{f\}(\vec{\alpha}) = y$ is a theorem; if no such y exists, $\{f\}(\vec{\alpha})$ is undefined. Then $\{f\}(\vec{\alpha}) = y$ is true iff it is a theorem. We say that f is an index of F if $F(\vec{\alpha}) = \{f\}(\vec{\alpha})$ for all $\vec{\alpha}$, and define a function to be recursive if it has an index. It is then clear that each of I–V is correct, and that the functions defined by I–V have the indices we have indicated.

If $\{f\}(\vec{\alpha}) = y$ is a true formula, the *ordinal* of $\{f\}(\vec{\alpha}) = y$ is the least σ such that $\{f\}(\vec{\alpha}) = y$ is in T_σ. Since y is uniquely determined by f and $\vec{\alpha}$, we also call σ the ordinal of $\{f\}(\vec{\alpha})$; so $\{f\}(\vec{\alpha})$ has an ordinal iff it is defined. If $\{f\}(\vec{\alpha}) = y$ has ordinal σ, then it is inferable from a subclass A of $\bigcup_{\tau<\sigma} T_\tau$; and an examination of the rules of inference shows that A is uniquely determined. The members of A are called the *predecessors* of $\{f\}(\vec{\alpha}) = y$, and the left sides of the members of A are called the *predecessors* of $\{f\}(\vec{\alpha})$. These predecessors then have ordinals $< \sigma$; this is useful in giving proofs by induction on σ.

The Enumeration Theorem is a consequence of our definition. Since we can define the S function much as before, we can prove the Recursion Theorem. This can be used to show that the old IV and V (extended to allow higher type arguments) hold in the present theory. The proof of IV is much like the application of Recursion Theorem to definitions by induction given in §2. (It is here that we need the functions Pr and DC.) To treat V, we use the Recursion Theorem to define a function H such that $H(y, \vec{\alpha}) = 0$ if $G(y, \vec{\alpha}) = 0$ and $H(y, \vec{\alpha}) = Sc(H(Sc(y), \vec{\alpha}))$ if $G(y, \vec{\alpha}) > 0$. We leave it to the reader to show that $\mu y(G(y, \vec{\alpha}) = 0) = H(0, \vec{\alpha})$. We can then develop the basic theory as in §2; but there is no Normal Form Theorem. Observe also that $\alpha^{t+1}(\beta^t)$ is a recursive function of α and β; this is easily proven by induction on t, using IV.

We would, of course, like to show that for arguments of type 0 and 1, the recursive functions given by the definition of this section are the same as those defined in §2 and §5. We shall consider only functions with arguments of type 0; when these are taken care of, the rest is simple. The above shows that the recursive functions of arguments of type 0 form a recursively closed class, and therefore include the recursive functions of §2. For the converse, one notes that when dealing only with functions of type 0 arguments, one can prove a Normal Form Theorem almost exactly as before. Of course, the T_n and U are different from the T_n and U in §2; but they are defined similarly and hence are recursive in the sense of §2. But then the Normal Form

Theorem shows that every function of type 0 variables which is recursive in the present sense is recursive in the sense of §2.

The Substitution Theorem as stated in §5 does not hold here, as the following example shows. Let A be a Π_1^0 set which is not Σ_1^0; say $x \in A \leftrightarrow \forall y R(x, y)$ with R recursive. Let $K(x, y) = \mu z R(x, y)$, so that $K(x, y) = 0$ if $R(x, y)$ and $K(x, y)$ is undefined otherwise. Then K is recursive and $\lambda y K(x, y)$ is a real iff $x \in A$. Define recursive G and H by $G(\alpha^{(2)}, x) = \alpha(\lambda y K(x, y))$, $H(x, \beta^{(1)}) = 0$, and let $F(x) = G(\lambda\beta H(x, \beta), x)$. Then $F(x)$ is defined iff $x \in A$. Since the domain of F is not RE, F is not recursive. There are various assumptions on G and H which make the Substitution Theorem true; the simplest is that G is total.

Substitution Theorem. *If G and H are recursive and total, and F is defined by $F(\vec{\alpha}) = G(\lambda\delta H(\vec{\alpha}, \delta), \vec{\alpha})$, then F is recursive.*

If $i = \langle n, i_1, \ldots, i_p\rangle$ where $1 \leq i_j \leq n$ for all j, we write $\pi_i(\alpha_1, \ldots, \alpha_n)$ for $(\alpha_{i_1}, \ldots, \alpha_{i_p})$. The Substitution Theorem is a consequence of the following.

Lemma. *For each $t \geq 1$, there is a recursive total function Z_t such that $\{Z_t(f, h, i)\}(\vec{\alpha}) = \{f\}(\pi_i(\lambda\delta^{t-1}\{h\}(\vec{\alpha}, \delta), \vec{\alpha}))$ provided the right side is defined.*

The proof is by induction on t. We let $i = \langle n, i_1, \ldots, i_p\rangle$ and write ε for $\lambda\delta\{h\}(\vec{\alpha}, \delta)$. We assume that Z_t is defined and prove $\{Z_t(f, h, i)\}(\vec{\alpha}) = \{f\}(\pi_i(\varepsilon, \vec{\alpha}))$ by induction on the ordinal of $\{f\}(\pi_i(\varepsilon, \vec{\alpha}))$, dividing into cases according to the nature of the index f. In each case, we find that Z_t must satisfy an equality whose left side is $Z_t(f, h, i)$, and whose right side may contain Z_t or an index of Z_t. We can then find a recursive Z_t which satisfies the equalities by the Recursion Theorem; and it will be total.

We shall examine one important case. Suppose $f = \langle 4, g\rangle$ and $i_1 = 1$. Then

$$\begin{aligned}
\{f\}(\pi_i(\varepsilon, \vec{\alpha})) &= \varepsilon(\lambda\gamma\{g\}(\pi_i(\varepsilon, \vec{\alpha}), \gamma)) \\
&= \{h\}(\vec{\alpha}, \lambda\gamma\{g\}(\pi_i(\varepsilon, \vec{\alpha}), \gamma)) \\
&= \{h\}(\vec{\alpha}, \lambda\gamma\{g\}(\pi_j(\varepsilon, \vec{\alpha}, \gamma)))
\end{aligned}$$

where j is a recursive function of i. Using II, we can find a recursive function h' of h such that $\{h'\}(\vec{\alpha}, \gamma, \delta) = \{h\}(\vec{\alpha}, \delta)$. Since $\{g\}(\pi_j(\varepsilon, \vec{\alpha}, \gamma))$ is a predecessor of $\{f\}(\pi_i(\varepsilon, \vec{\alpha}))$, the induction hypothesis shows that $\{g\}(\pi_j(\varepsilon, \vec{\alpha}, \gamma)) = \{Z_t(g, h', j)\}(\vec{\alpha}, \gamma)$, and hence

$$\{f\}(\pi_i(\varepsilon, \vec{\alpha})) = \{h\}(\vec{\alpha}, \lambda\gamma\{Z_t(g, h', j)\}(\vec{\alpha}, \gamma)).$$

Since ε is of type t, γ is of type $t - 2$. Hence by the induction hypothesis on t,

$$\{h\}(\vec{\alpha}, \lambda\gamma\{Z_t(g, h', j)\}(\vec{\alpha}, \gamma)) = \{Z_{t-1}(h, Z_t(g, h', j), k)\}(\vec{\alpha})$$

where k is a recursive function of i. Thus we want $Z_t(f, h, i) = Z_{t-1}(h, Z_t(g, h', j), k)$.

By induction on t, we define a mapping *up* of the set of type t objects into the set of type $t+1$ objects and a mapping *do* of the set of type $t+1$ objects into the set of type t objects such that $do(up(\alpha)) = \alpha$ for all α (and hence *up* is one-one and *do* is onto). For $t = 0$, we let $up(x) = \lambda yx$ and $do(\alpha) = \alpha(0)$. For $t > 0$, let $up(\alpha^{(t)}) = \lambda\beta^{(t)}\alpha(do(\beta))$ and $do(\alpha^{(t+1)}) = \lambda\beta^{(t-1)}\alpha(up(\beta))$. By the Substitution Theorem, we see that *up* and *do* can be used in definitions of total recursive functions.

Now we shall see how to assign codes to finite sequences of objects. For $t > 0$, we define a type t object $\langle\alpha_1, \dots, \alpha_n\rangle^t$ whenever $\alpha_1, \dots, \alpha_n$ all have type $\leq t$. Let β_i be the type t object obtained by applying *up* repeatedly to α_i. Then $\langle\alpha_1, \dots, \alpha_n\rangle^t = \lambda\gamma^{t-1}\langle\beta_1(\gamma), \dots, \beta_n(\gamma)\rangle$. For $j \leq t$, let $(\alpha^t)_i^j$ be the object of type j obtained from $\lambda\beta(\alpha(\beta))_i$ by repeatedly applying *do*. Then for α_i of type t_i and $t \geq t_i$ for all $i < n$, $(\langle\alpha_0, \dots, \alpha_{n-1}\rangle^t)_i^{t_i} = \alpha_i$. All of these functions can be used in definitions of total recursive functions.

We can use this coding of finite sequences to define the contraction of a total function F as before. If the arguments of F have types $t_1, \dots, t_n$, then $\langle F\rangle$ is an object of type $1 + \max t_i$; and F is recursive iff $\langle F\rangle$ is recursive. We can then define relative recursion by using the alternate definition of §5: F is recursive in Φ if there are $G_1, \dots, G_n \in \Phi$ and a recursive F' such that

$$F(\vec{\alpha}) = F'(\langle G_1\rangle, \dots, \langle G_n\rangle, \vec{\alpha})$$

for all $\vec{\alpha}$.

Now let us turn to extensions of the analytical hierarchy. For $r \geq 1$, a relation is of *order* $r+1$ if it has a definition consisting of quantifiers of type $\leq r$ followed by a recursive relation. Much as before, we can suppose that in the prefix no two successive quantifiers are of the same kind, and that all of the quantifiers are of type r except the last, which is of type $r-1$. If the prefix has k quantifiers of type r and the first is existential { universal }, the relation is $\Sigma_k^r\{\Pi_k^r\}$. A relation is Δ_k^r if it is both Σ_k^r and Π_k^r.

As before, every relation of order r is Σ_k^r and Π_k^r for all k, and every Σ_k^r or Π_k^r relation is Σ_n^r and Π_n^r for all $n > k$. Again we have closure properties; these show that the position of a relation in the order $r+1$ hierarchy is not changed by prefixing quantifiers of type $< r$.

We again define a relation to be RE if it is the domain of a recursive function. Again Σ_1^0 relations are RE; but the converse is no longer true. The best we can prove is the following.

THEOREM. *Let $r \geq 1$, and let R be an RE relation having arguments of type $\leq r+1$. Then R is Π_1^1 if $r = 1$ and Δ_1^r if $r > 1$.*

Let us see how this theorem gives us the Enumeration Theorem for Σ_k^r and Π_k^r provided that all the arguments of the functions have type $\leq r+1$. (The Enumeration Theorem is false without this provision.) It is enough to treat the Π_1^r case. Let all the variables in $\vec{\alpha}$ have type $\leq r+1$ and define

$$P(e, \vec{\alpha}) \leftrightarrow \forall \beta^r \exists \gamma^{r-1} W_e(\vec{\alpha}, \beta, \gamma).$$

Now $W_e(\vec{\alpha}, \beta, \gamma)$, as a relation of $e, \vec{\alpha}, \beta, \gamma$, is the domain of the enumerating function $\{e\}(\vec{\alpha}, \beta, \gamma)$; so it is RE and hence Π_1^r. By the closure conditions, P is Π_1^r; so each $P_{(e)}$ is Π_1^r. If Q is a Π_1^r relation of $\vec{\alpha}$, then $Q(\vec{\alpha}) \leftrightarrow \forall \beta \exists \gamma R(\vec{\alpha}, \beta, \gamma)$ with R recursive. Then R is Σ_1^0 and hence RE. If e is an RE index of R, then $Q = P_{(e)}$.

Now we outline the proof of the theorem. (We have incorporated some simplifications of Kleene's proof due to Platek [8].) Fix r, and let $\vec{\alpha}$ be a sequence of objects having types $\leq r+1$. An *$\vec{\alpha}$-formula* is a formula $\{f\}(\vec{\beta}) = y$ where every object in $\vec{\beta}$ is either in $\vec{\alpha}$ or of type $< r$. Of course, $\{f\}(\vec{\alpha}) = y$ is an $\vec{\alpha}$-formula; and it is easy to see that every predecessor of a true $\vec{\alpha}$-formula is an $\vec{\alpha}$-formula. We can code the $\vec{\alpha}$-formulas by type $r-1$ objects as follows. Let $\vec{\alpha}$ be $\alpha_0, \dots, \alpha_{n-1}$. If β_i is α_j, γ_i is j; if β_i is a number z, $\gamma_i = n+z$; otherwise, $\gamma_i = \beta_i$. Then the code of $\{f\}(\vec{\alpha}) = y$ is $\langle f, \langle \vec{\gamma} \rangle, y \rangle$. Let $Tr(\vec{\alpha}, \beta)$ mean that β is the code of a true $\vec{\alpha}$-formula. We shall prove that Tr is Π_1^1 if $r = 1$ and Δ_1^r if $r > 1$. Then if f is an RE-index of R,

$$R(\vec{\alpha}) \leftrightarrow \exists y\, Tr(\vec{\alpha}, \langle f, \langle 0, \dots, n-1 \rangle, y \rangle);$$

so R is Π_1^1 if $r = 1$ and Δ_1^r if $r > 1$.

Let $Inf(\vec{\alpha}, \beta^r, \gamma^{r-1})$ mean that β is the set of codes of a regular class A of $\vec{\alpha}$-formulas and that γ is the code of an $\vec{\alpha}$-formula inferable from formulas in A. Then Inf is of order r. To prove this, we write out a definition of Inf and check that all quantifiers are of type $\leq r-1$. The only problem arises when γ is the code of a formula $\{\langle 4, g \rangle\}(\alpha_i, \vec{\delta}) = x$ where α_i is of type $r+1$. We must say that $x = \alpha_i(\iota)$ where for each v, $\{g\}(\alpha_i, \vec{\delta}, v) = \iota(v)$ is in A. Then ι is of type r and v of type $r-1$; so we may quantify over v but not over ι. This is all right, because regularity tells us that $\iota = \lambda \vec{v} \mu y (\beta(\dots) = 0)$ where $\dots$ is the code of $\{g\}(\alpha_i, \vec{\delta}, v) = y$.

A class A of $\vec{\alpha}$-formulas is *$\vec{\alpha}$-closed* if it is regular and every $\vec{\alpha}$-formula inferable from formulas in A is in A. The class of true $\vec{\alpha}$-formulas is clearly $\vec{\alpha}$-closed. Moreover, it is the smallest $\vec{\alpha}$-closed class. To see this, we must show that if A is $\vec{\alpha}$-closed, then every true $\vec{\alpha}$-formula a is in A; and we do this by induction on the ordinal of a. The predecessors of a are true $\vec{\alpha}$-formulas and hence in A; so $a \in A$ because A is $\vec{\alpha}$-closed.

Let $Cl(\vec{\alpha}, \gamma^r)$ mean that γ is the set of codes of an $\vec{\alpha}$-closed class. Since Inf is of order r, Cl is of order r. By the above paragraph,

$$Tr(\vec{\alpha}, \beta) \leftrightarrow \forall \gamma^r (Cl(\vec{\alpha}, \gamma) \rightarrow \gamma(\beta) = 0).$$

Hence Tr is Π^r_1.

It remains to show that Tr is Σ^r_1 if $r > 1$. We recall that if R is a binary relation on a class A, then R *well-founds* A if there is no sequence $\{a_n\}$ in A which is R-*descending*, i.e., such that $R(a_{n+1}, a_n)$ for all n. Suppose that a is a true $\vec{\alpha}$-formula; A is the regular class of true $\vec{\alpha}$-formulas; and $R(b, c)$ iff $b, c \in A$ and the ordinal of b is less than the ordinal of c. Then: (i) $a \in A$; (ii) R well-founds A; (iii) for all $b \in A$, b is inferable from formulas in $\{c : R(c, b)\}$. Conversely, if a is an $\vec{\alpha}$-formula, A a regular class of $\vec{\alpha}$-formulas, R a binary relation on A and (i)–(iii) hold, then a is true; for otherwise, we can use (i) and (iii) to construct an R-descending sequence of false formulas in A, contradicting (ii).

It follows that if A varies through sets of type $r - 1$ objects and R through binary relations on type $r - 1$ objects,

$$Tr(\vec{\alpha}, \beta) \leftrightarrow$$
$$\exists A \exists R[\beta \in A \,\&\, R \text{ well-founds } A \,\&\, (\forall \gamma \in A)(\mathit{Inf}(\vec{\alpha}, \{\delta : R(\delta, \gamma)\}, \gamma)].$$

Thus it suffices to show that 'R well-founds A' is a relation of A and $\langle R\rangle$ of order r when $r > 1$. For γ of type $r - 1$, define $[\gamma]_i = \lambda\delta^{r-2}\gamma(\langle i, \delta\rangle)$. Then for each infinite sequence $\{\gamma_i\}$ of type $r - 1$ objects, there is a γ such that $\gamma_i = [\gamma]_i$ for all i; it is defined by $\gamma(\delta) = \gamma_i((\delta)^{t-1}_1)$ where $i = (\delta)^0_0$. Then

$$R \text{ well-founds } A \leftrightarrow \forall \gamma^{r-1} \neg \forall n([\gamma]_n \in A \,\&\, R([\gamma]_{Sc(n)}, [\gamma]_n)).$$

A particularly interesting type 2 object is the object $\mathbf{E}$ defined by $\mathbf{E}(\alpha) = 0$ if $\exists x(\alpha(x) = 0)$, $\mathbf{E}(\alpha) = 1$ otherwise. We can reasonably identify this object with the existential number quantifier. It is easy to see that every arithmetical relation is recursive in $\mathbf{E}$. More generally:

THEOREM. *A set is recursive in* $\mathbf{E}$ *iff it is hyperarithmetical.*

First, every set which is RE in $\mathbf{E}$ is Π^1_1. The proof of this is practically the same as the above proof with $r = 1$, $\vec{\alpha} = \mathbf{E}$. The only place where the definition of $\mathbf{E}$ is pertinent is in the proof that *Inf* is of order r,i.e., arithmetical. This is all right, since, $\mathbf{E}(\alpha) = y$ is clearly arithmetical.

If A is recursive in $\mathbf{E}$, then A and $\neg A$ are RE in $\mathbf{E}$ and hence Π^1_1; so A is Δ^1_1 and hence hyperarithmetical. To prove the converse, it sufficed to show that each H_σ is recursive in $\mathbf{E}$. For this, we define a recursive G so that if $e \in O$, $H_{|e|} = \lambda x G(\mathbf{E}, e, x)$. Assume that G is defined, and let us prove this by induction on $|e|$. By (3) of §5 and the induction hypothesis,

$$H_{|e|}(x) \leftrightarrow \exists f (f <_O e \,\&\, \neg W_x(\lambda y\{g\}(\mathbf{E}, f, y))$$

where g is an index of G. The right side is arithmetical and hence is a recursive relation R of $g, \mathbf{E}, e, x$. Thus we need to have $G(\alpha^2, e, x) = R(g, \alpha, e, x)$, which we can accomplish by the Recursion Theorem.

A great deal of work in recursion in higher types has been done since the basic papers; but we shall only mention one point which continued to interest Kleene. This was the failure of the Substitution Theorem (in the strong form of §5). It is clear that the difficulty is that the right side of the equation is necessarily undefined if $\lambda\delta H(\vec{\alpha}, \delta)$ is not total. Thus Kleene suggested in *1963* that one should allow objects to be partial. An object of type $t + 1$ is to be a partial mapping of the set of objects of type t into the set of numbers. However, if $t \neq 0$, we do not take all such mappings. We require that if $\alpha^{t+1}(\beta) = x$, and β' is an extension of β, then $\alpha(\beta') = x$. These objects are called *monotone* objects.

The first person to develop recursion theory using monotone objects was Platek in his thesis [8]. He also used another idea first suggested in *1963*; instead of building the Enumeration Theorem into the definition, he built in the First Recursion Theorem, which is a form of the Recursion Theorem which does not involve indices.[16] This means that the recursive functions could be introduced directly by the schemes as in §2, instead of the indirect method used here. Platek actually used a great many more types (e.g., types of n-ary functions) so that there was no longer any difference between objects and functions. The result was an extremely elegant definition of the class of recursive functions. More or less the same theory, formulated more along the lines of Kleene, was discovered by Kleene's student Kierstead [4].

Kleene felt that the proper way to select the objects was by consideration of how an oracle for the object should behave. To him, this justified monotoneness; for if the oracle for α decides that $\alpha(\beta) = x$ from being given information about β, and later new information about β is discovered, then the oracle should not change her decision. (Oracles do not make mistakes!) In a series of papers[17] (*1978*, *1980*, *1982*, *1984*), Kleene considered more closely how oracles behave and arrived at the more restrictive class of *unimonotone* objects. Perhaps because Kleene never got around to developing the recursion theory of these objects, it is not clear that the much more complicated unimonotone objects have any advantage over the monotone objects.

§8. Miscellaneous. There are two other articles of Kleene which deserve at least a brief discussion here. In *1952a*, he considers a problem from logic: when is a first order theory finitely axiomatizable? (We assume that

[16] Kleene himself seemed quite pleased with the idea of using the First Recursion Theorem to define the class of recursive functions. In *1981b*, he defines the recursive functions of §5 in this manner.

[17] These articles show Kleene's sense of humor in a way that earlier articles never did. For example, his oracles are feminine. He said this was because it was the most recent practice about oracles; but I suspect it was his bow to the feminist movement. I might also mention the obscure dedication of *1978*, which is only explained in the final footnote.

all first order theories have only finitely many nonlogical constants.) This problem has inspired a lot of work by model theorists, without receiving a fully satisfactory answer.

Kleene considered a modified problem: when is a theory finitely axiomatizable using additional constants? In other words, when does a theory T have a finitely axiomatizable extension T' such that a sentence in the language of T is provable in T' iff it is provable in T?

We assume that the formulas of each theory have been assigned codes in a reasonable way. We may then apply concepts of recursion theory by replacing expressions by their codes. Now if T and T' are as above, then the set of theorems of T' is RE; the proof is just like the proof in §3 that the set of codes of true formulas is RE. It follows that the set of theorems of T is RE. Kleene proved that this necessary condition for finite axiomatizability using additional constants is also sufficient.

Suppose that the set of theorems of T is an RE set A. We can construct a finitely axiomatized subsystem S of arithmetic with a formula $\phi(x)$ such that for each number k, $\phi(k)$ is provable in S iff $k \in A$. (This construction essentially goes back to Church in his proof of the nonexistence of algorithm result mentioned in §1.) We get T' from S by introducing the symbols of T and the unary predicate symbol N; the axiom $\phi(x) \to N(x)$; and a finite number of axioms which insure that if k is the code of a sentence ψ, then $\psi \leftrightarrow N(k)$ is a theorem. It is then clear that every theorem of T is a theorem of T'. Now suppose that χ is a sentence in T which is a theorem of T'. If χ is not a theorem of T, we can find a model of T having the set of numbers as its universe in which χ is false. It will be enough to extend this to a model of T'. The symbols of S have obvious meanings which makes the axioms of S true. The meaning of N is chosen to make the axioms used to prove $\psi \leftrightarrow N(k)$ true. The axiom $\phi(k) \to N(k)$ is true because if $\phi(k)$ is true, then k is the code of a theorem ψ of T, and hence ψ and $\psi \leftrightarrow N(k)$ are true in the model.

At the time, Kleene was much interested in intuitionism; so he wanted to show that his result held also in intuitionistic systems. Here we have no completeness theorem; so the last part of the proof must show directly how to convert a proof of χ in T' into a proof in T. This can be done by using Gentzen proof theory; but it requires supplementing Gentzen's Haupsatz by a result saying that we can always use Gentzen's rules in a certain order. The proof of this result (which is of general interest in proof theory) is the other paper in *1952a*.

In the summer of 1951, Kleene went to work at RAND. At someone's suggestion, he read an article by McCulloch and Pitts, who were trying to build a model of neural activity based on a simple kind of machine. Kleene abstracted from their ideas the notion of a finite automaton, and proceeded to describe exactly what sets were defined by such automata.

Let Ω be a finite set of symbols. An Ω-*word* is a finite sequence of symbols in Ω. A *finite automaton* over Ω consists of the following things: (a) a finite set S of *states*; (b) a σ_0 in S, called the *initial* state; (c) a mapping T of $S \times \Omega$ into S, called the *transition function*; (d) a subset S_0 of S, called the set of *designated states*. We start the automaton by putting the machine in its initial state and putting a tape containing an Ω-word into the machine with its first symbol scanned. If at any time, the machine is in state σ and is scanning the symbol a, then it passes to the state $T(\sigma, a)$ and scans the next symbol if there is one; if the scanned symbol is the last symbol of the Ω-word, the machine stops. The machine *accepts* the Ω-word if it is in a designated state when it stops.

If A and B are Ω-words, $A \cdot B = \{ab \ : \ a \in A \,\&\, b \in B\}$. We define A^n inductively by $A^0 = \{\emptyset\}$ (where $\emptyset$ is the empty Ω-word), $A^{n+1} = A^n \cdot A$. Let $A^* = \bigcup_n A^n$. Let $\mathcal{R}$ be the smallest class of sets of Ω-words such that: (i) the empty set is in $\mathcal{R}$; (ii) $\{\emptyset\}$ is in $\mathcal{R}$; (iii) if a is a symbol, then $\{a\}$ is in $\mathcal{R}$; (iv) if A and B are in $\mathcal{R}$, then $A \cup B$ and $A \cdot B$ are in $\mathcal{R}$; (v) if A is in $\mathcal{R}$, then A^* is in $\mathcal{R}$. Kleene's main result was that a set of Ω-words is the set accepted by a finite automaton iff it is in $\mathcal{R}$.

After writing up this proof in *1956a*, Kleene gave no more attention to finite automaton. However, his theorem has proved fundamental in all further research in the subject.

Besides research, Kleene did a great deal of expository writing, some of it aimed at nonlogicians or even nonmathematicians. His writing is not notable for elegance or conciseness; but it is always clear and precise, and tells the readers what he really needs to know. His most important expository work is *1952b*, which, as we have indicated, is also an important research work. It arrived too late for my education; I learned most of my basic logic from the original paper-covered edition of Church's ***Introduction to Mathematical Logic***, which I still consider one of the gems of the literature of logic. However, the logicians who followed me usually learned their logic from *1952b*. Gerald Sacks and Michael Morley (among others) have said they read the book word by word during their graduate careers. In looking back at *1952b* in preparation for writing this article, I was surprised how much of it seems pertinent today.

Kleene was not what one would call a brilliant teacher; but he worked hard at teaching and always did a good job. His list of Ph.D. students cannot match those of Church, Tarski, or Sacks; but it is quite respectable nevertheless. Although he would never consider writing a thesis for a student, he was willing to spend large amounts of time in encouraging and advising his students. It seems fair to say that each of them wrote the best thesis of which he was capable.

Kleene was a great supporter of the ASL, and served a term as its president. He almost single-handedly ran the research section of the JSL for many years (while Church ran the reviews).

Of course, mathematics was the central interest of Kleene's professional life; but he had many other interests. I recall him taking part in noncurricular activities at meetings, from the swimming at the July 4th picnic at the 1957 Cornell meeting to rock climbing at the Kleene Symposium in Madison in 1978.

Kleene was, of course, respected by several generations of logicians; but most of them also looked on him as a friend. He always treated the logicians he met well, irrespective of their abilities or reputations. I first met him in 1956[18], when I was a NSF Postgraduate Fellow at the Institute with very little published; he treated me as kindly then as in our meetings in later years, when I was better known in logical circles.

Stephen Cole Kleene died in Madison, Wisconsin on January 25, 1994.

REFERENCES

KLEENE BIBLIOGRAPHY

A bibliography of Kleene's work through 1980 is contained in the volume ***The Kleene Symposium*** [2], which also contains a complete list of his Ph.D. students and a brief biography. This bibliography lists only research articles and expository articles written for mathematicians.

[*1934*] *Proof by cases in formal logic*, ***Annals of Mathematics***, vol. 35 (1934), pp. 529–544.

[*1935a*] *A theory of positive integers in formal logic*, ***American Journal of Mathematics***, vol. 57 (1935), pp. 153–173, 219–244.

[*1935b*] *The inconsistency of certain formal logics*, with J. B. Rosser, ***Annals of Mathematics***, vol. 36 (1935), pp. 630–636.

[*1936a*] *General recursive functions of natural numbers*, ***Mathmatische Annalen***, vol. 112 (1936), pp. 727–742.

[*1936b*] *λ-definability and recursiveness*, ***Duke Mathematical Journal***, vol. 2 (1936), pp. 340–353.

[*1936c*] *A note on recursive functions*, ***Bulletin of the American Mathematical Society***, vol. 42 (1936), pp. 544–546.

[18] I recall being paired with him for the logic treasure hunt at John Addison's Thanksgiving party that year. John later confessed that he paired us because he wanted to know if I could keep up with the strides of Kleene, who was well over six feet and reasonably athletic. I must have managed, since we found the prize, which was a bottle of white wine. John then announced that the entire wine budget has been spent on that bottle, and suggested we contribute it to the Thanksgiving dinner. Those were lean years economically for mathematicians.

[*1936d*] *Formal definitions in the theory of ordinal numbers*, with ALONZO CHURCH, ***Fundamenta Mathematica***, vol. 28 (1936), pp. 11–21.

[*1938*] *On notation for ordinal numbers*, ***Journal of Symbolic Logic***, vol. 3 (1938), pp. 150–155.

[*1939*] *A postulational basis for probability*, with H. P. EVANS, ***American Mathematical Monthly***, vol. 46 (1939), pp. 141–148.

[*1943*] *Recursive predicates and quantifiers*, ***Transactions of the American Mathematical Society***, vol. 53 (1943), pp. 41–73.

[*1944*] *On the form of the predicates in the theory of constructive ordinals*, ***American Journal of Mathematics***, vol. 66 (1944), pp. 41–58.

[*1945*] *On the interpretation of intuitionistic number theory*, ***Journal of Symbolic Logic***, vol. 10 (1945), pp. 109–124.

[*1947*] *Analysis of lengthening of modulated repetitive pulses*, ***Proceedings of the Institute of Radio Engineers***, vol. 35 (1947), pp. 1049–1053.

[*1950*] *A symmetric form of Gödel's theorem*, ***Indagationes Mathematicae***, vol. 12 (1950), pp. 244–246.

[*1952a*] ***Two papers on the predicate calculus***, Memoirs of the American Mathematical Society, no. 10, Americal Mathematical Society, 1952.

[*1952b*] ***Introduction to metamathematics***, North-Holland Publishing Co., 1952.

[*1954*] *The upper semi-lattice of degrees of unsolvability*, with EMIL L. POST, ***Annals of Mathematics***, vol. 59 (1954), no. 2, pp. 379–407.

[*1955a*] *Arithmetical predicates and function quantifiers*, ***Transactions of the American Mathematical Society***, vol. 79 (1955), pp. 312–340.

[*1955b*] *On the form of the predicates in the theory of constructive ordinals (second paper)*, ***American Journal of Mathematics***, vol. 77 (1955), pp. 405–428.

[*1955c*] *Hierarchies of number-theoretic predicates*, ***Bulletin of the American Mathematical Society***, vol. 79 (1955), pp. 312–340.

[*1956a*] *Representation of events in nerve nets and finite automata*, ***Automata studies***, Annals of Mathematical Studies, no. 34, Princeton University Press, 1956, pp. 3–41.

[*1956b*] *A note on computable functionals*, ***Indagationes Mathematicae***, vol. 18 (1956), pp. 275–280.

[*1957*] *A note on function quantification*, with JOHN ADDISON, ***Proceedings of the American Mathematical Society***, vol. 8 (1957), pp. 1002–1006.

[*1958*] *Extension of an effectively generated class of functions by enumeration*, ***Colloquium Mathematicum***, vol. 6 (1958), pp. 67–78.

[*1959a*] *Recursive functionals and quantifiers of finite types I*, ***Transactions of the American Mathematical Society***, vol. 91 (1959), pp. 1–52.

[*1959b*] *Countable functionals*, ***Constructivity in mathematics, proceedings of the colloquium held in Amsterdam 1957***, North-Holland Publishing Co., 1959, pp. 81–100.

[*1959c*] *Quantifiers of number-theoretic functions*, ***Compositio Mathematica***, vol. 14 (1959), pp. 23–40.

[*1960*] *Realizability and Shanin's algorithm for the constructive deciphering of mathematical sentences*, ***Logique et Analyse***, vol. 3 (1960), pp. 154–165.

[*1962a*] *Disjunction and existence under implication in elementary intuitionistic formalisms*, ***Journal of Symbolic Logic***, vol. 27 (1962), pp. 11–18, Addendum, *ibid.*, vol. 28 (1963), pp. 154–156.

[*1962b*] *Turing-machine computable functionals of finite types I*, ***Logic, methodology, and philosophy of science: Proceedings of the 1960 international congress***, Stanford University Press, 1962, pp. 38–45.

[*1962c*] *Turing-machine computable functionals of finite types II*, ***Proceedings of the London***

Mathematical Society, vol. 12 (1962), no. 3, pp. 245–258.

[*1962d*] *Lambda-definable functionals of finite type*, ***Fundamenta Mathematicae***, vol. 50 (1962), pp. 281–303.

[*1962e*] *Herbrand-Gödel-style recursive functionals of finite types*, ***Recursive function theory***, Proceedings of the Symposia in Pure Mathematics, vol. 5, American Mathematical Society, 1962, pp. 49–75.

[*1963*] *Recursive functionals and quantifiers of finite type II*, ***Transactions of the American Mathematical Society***, vol. 108 (1963), pp. 106–142.

[*1965*] ***The foundations of intuitionistic matehmatics, especially in relation to recursive functions***, with RICHARD VESLEY, North Holland Publidhing Company, 1965.

[*1967*] ***Mathematical logic***, Wiley, 1967, 398pp.

[*1969*] ***Formalized recursive functionals and formalized realizability***, Memoirs of the American Mathematical Society, no. 89, Americal Mathematical Society, 1969.

[*1973*] *Realizability: a retrospective survey*, ***Cambridge summer school in mathematical logic***, Lecture Notes in Mathematics, no. 337, Springer-Verlag, 1973, pp. 95–112.

[*1976*] *The work of Kurt Gödel*, ***Journal of Symbolic Logic***, vol. 41 (1976), pp. 761–778, Addendum, *ibid.*, vol. 43 (1978), p. 613.

[*1978*] *Recursive functionals and quantifiers of finite type revisited I*, ***Generalized recursion theory II: Proceedings of the 1977 Oslo symposium***, North-Holland Publishing Co., 1978, pp. 185–222.

[*1980*] *Recursive functionals and quantifiers of finite type revisisted II*, ***The Kleene symposium***, Studies in logic and the foundations of mathematics, no. 101, North-Holland Publishing Co., 1980, pp. 1–29.

[*1981a*] *Origins of recursive function theory*, ***Annals of the History of Computing***, vol. 3 (1981), pp. 52–65.

[*1981b*] *The theory of recursive functions, approaching its centennial*, ***Bulletin of the American Mathematical Society***, vol. 5 (1981), pp. 43–61.

[*1982*] *Recursive functionals and quantifiers of finite type revisited III*, ***Patras logic symposium***, Studies in logic and foundations of mathematics, no. 109, North-Holland Publishing Co., 1982, pp. 1–40.

[*1984*] *Unimonotone functions of finite type (recursive functionals and quantifiers of finite type revisited IV)*, ***Recursion theory***, Proceedings of Symposia in Pure MathematicsN, vol. 42, American Mathematical Society, 1984, pp. 119–138.

[*1987*] *Reflections on Church's Thesis*, ***Notre Dame Journal of Formal Logic***, vol. 28 (1987), pp. 490–498.

SUPPLEMENTARY BIBLIOGRAPHY

[1] J. W. ADDISON, *Separation principles in the hierarchies of classical and effective descriptive set theory*, ***Fundamenta Mathematicae***, vol. 46 (1959), pp. 123–135.

[2] Jon Barwise, H. Jerome Keisler, and Kenneth Kunen (editors), ***The Kleene symposium***, Studies in Logic, vol. 101, North Holland Publishing Company, Amsterdam, New York, Oxford, 1980.

[3] ROGER HINDLEY and JONATHAN SELDIN, ***Introduction to combinators and λ-calculus***, Cambridge University Press, 1986.

[4] DAVID KIERSTEAD, *Syntax and semantics in higher-type recursion theory*, ***Transactions of the American Mathematical Society***, vol. 276 (1983), pp. 67–105.

[5] JOAN RAND MOSCHOVAKIS, *Disjunction and existence in formalized intuitionistic analysis*, ***Proceedings of the summer school in mathematical logic, Leicester***, North-Holland Publishing Co., 1967, pp. 309–331.

[6] David Nelson, *Recursive functions and intuitionistic number theory*, ***Transactions of the American Mathematical Society***, vol. 61 (1967), pp. 307–368.

[7] David Normann, ***Recursion on countable functionals***, Lecture Notes in Mathematics, no. 811, Springer-Verlag, 1980.

[8] Richard Platek, *Foundations of recursion theory*, ***Ph.D. thesis***, Stanford University, 1966.

[9] Emil Post, *Recursively enumerable sets of positive integers and their decision problem*, ***Bulletin of the American Mathematical Society***, vol. 50 (1944), pp. 284–316.

[10] Robert Soare, ***Recursively enumerable sets and degrees***, Springer-Verlag, 1987.

[11] Clifford Spector, *Recursive well-orderings*, ***Journal of Symbolic Logic***, vol. 20 (1955), pp. 151–163.

[12] ———, *On degrees of recursive unsolvability*, ***Annals of Mathematics***, vol. 64 (1956), no. 2, pp. 581–592.

[13] Alan Turing, *Computability and λ-definability*, ***Journal of Symbolic Logic***, vol. 2 (1937), pp. 153–163.

DEPARTMENT OF MATHEMATICS
DUKE UNIVERSITY
DURHAM, NC 27706

E-mail: jrs@math.duke.edu