But even if the intellect is not simply a material system, we would still want to know how it works. However, we must again be careful. It sometimes makes sense to ask "how" something works, in the sense of seeking a "mechanism" by which it happens, but sometimes it does not. For example, one can answer the question of how a phonograph produces sound, or how a television produces a picture. But it is not clear that it makes sense to ask "how" a mass produces a gravitational field, say. In Newton's theory, "mass" and "gravitational field" are fundamental concepts. Newton's law of gravitation posits the existence of a relationship between them and gives a quantitative account of that relationship. But it does not explain "how" the mass produces the field. As Newton himself said in the famous concluding words of his *Principia*: "I have not been able to discover the cause of those properties of gravity from phenomena, and I frame no hypotheses. . . . [It] is enough that gravity does really exist and act according to the laws which we have explained, and abundantly serves to account for all the motions of the celestial bodies, and of our sea."[25] Similarly, Einstein's theory, while it gives a deeper understanding of what a gravitational field is, namely the curving of space-time, does not explain "how" a massive body causes that curvature, in the sense of a mechanism.

All we know about the human intellect is that it is capable of having insights, of understanding meanings. By what mechanism? "How" does the intellect act on the physical brain? What is the intellect made of? Far from requiring a materialist answer, these questions may not even turn out to make any sense. Sometimes we must have the patience to hold certain questions in abeyance until we have the conceptual equipment and level of understanding that allows us to distinguish the good questions from the bad ones. To repeat again the wise words of Hermann Weyl: "One of the great differences between the scientist and the impatient philosopher is that the scientist bides his time. We must await the further development of science, perhaps for centuries, perhaps for thousands of years, before we can design a true and detailed picture of the interwoven texture of Matter, Life, and Soul."

22 Is the Human Mind Just a Computer?

What a Computer Does

As we saw in the last chapter, the materialist conceives of the human mind as being no more than a computer in operation. In this chapter I am going to explain some arguments, due to the philosopher John R. Lucas and the mathematician Roger Penrose, and based on a powerful theorem proved by the logician Kurt Gödel, that attempt to show that this conception cannot be right.

What is a computer? The name itself tells us: it is a device that computes. Computers come in two types, analog and digital. The difference is somewhat like the difference between an hourglass and a digital clock. The hourglass measures time by using a physical or mechanical process, while the digital clock reduces time to numbers. At one time, very sophisticated analog computers were built—for example, they were used to aim the huge guns on warships of World War II vintage. However, since the advent of electronics, almost all computers are digital. For the present discussions we can restrict our attention to digital computers, since the issue which concerns us has to do with what computers are capable of doing, and anything which could be done by an analog computer can be simulated by a digital computer.

Digital computers manipulate numbers. The numbers they display for us are usually in the decimal, or base-10, notation that we are taught in school. This is true of the displays of most pocket calculators, for instance. The numbers that computers use internally, however, are generally in binary, or base-2, form. That is, instead of using the digits 0 through 9, they use the two "bits" or "binary digits" 0 and 1. These are not actually written out as "0" and "1" inside the computer, of course; rather, 0 and 1 may be represented by some voltage having one value or another, or by some small piece of material being magnetized or not magnetized. This illustrates an important point: computers deal with information that is expressed in some kind of symbols, and it does not matter what particular symbols are used.

Rather than saying that computers manipulate numbers, then, it would be more accurate to say that they manipulate symbols. The symbols could stand for numbers, but they could also stand for words, or moves in a chess game, or even nothing at all. Computers manipulate these symbols using a rote procedure, called a "program" or "algorithm." At every step of its operations, the program or algorithm tells the computer exactly which manipulations to perform.

The idea of an algorithm may seem foreign to those who have not worked with computers, but actually almost everyone is familiar with certain simple algorithms. The procedures we all learned in grade school for doing addition, subtraction, multiplication, and long-division are algorithms. In doing a long-division problem, for example, we do a sequence of simple steps involving the symbols 0 through 9 in accordance with a well-defined recipe. The "inputs" are the numbers being divided, and the "output" is their quotient.

Now, for obvious reasons, mathematicians are very interested in this business of manipulating symbols using algorithms. Not only can mathematical calculations be done in this way, as we just saw, but so can mathematical proofs. In a proof one starts with certain statements that are assumed to be true—these are either unproven "axioms" or theorems that have been proven previously—and derives from them some new theorem using certain "rules of inference." A rule of inference is a logical or mathematical rule that enables one to deduce statements from other statements. All of this can be done by computers. First, the axioms and theorems have to be expressed in some symbolic language as strings of symbols that the computer can manipulate, and then the rules of inference have to be reduced to precise recipes for manipulating strings of symbols to produce new strings of symbols.

Let us give a simple illustration. Suppose that one of the axioms is that $x = x$, where x stands for any string of symbols. This axiom would tell us, for example, that $2 = 2$, and $17 = 17$, and $a + 2 = a + 2$ are true statements. And suppose that one of the rules of inference is that $x + y$ on one side of an = sign can be replaced by $y + x$, where x and y are strings of symbols.[1] For example, this rule of inference tells us that if we see $2 + 5$ in a true statement, we may replace it with $5 + 2$ and the statement will remain true. Similarly, we can replace $a + b$ with $b + a$. Now suppose that the theorem we are trying to prove is that $a + b + c = c + b + a$. One way to proceed (but not the shortest) is to take the following series of steps. Start with

$$a + b + c = a + b + c.$$

This is a true statement because of the axiom $x = x$, where x in the axiom is taken to be $a + b + c$. Next, on the right-hand side of the =, replace $b + c$ with $c + b$ (an application of the rule of inference):

$$a + b + c = a + \underset{\text{(switch)}}{\underline{b + c}}.$$

This gives

$$a + b + c = a + c + b.$$

Then, use the rule of inference again to replace $a + c$ on the right-hand side of the = with $c + a$:

$$a + b + c = \underset{\text{(switch)}}{\underline{a + c}} + b.$$

This gives

$$a + b + c = c + a + b.$$

Finally, replace $a + b$ with $b + a$ on the right:

$$a + b + c = c + \underset{\text{(switch)}}{\underline{a + b}}.$$

This gives the final result

$$a + b + c = c + b + a,$$

which is the result that we set out to prove.

This is a very simple example, but in fact even difficult proofs in mathematics can be done by such mechanical methods of symbol manipulation. What this indicates is that at least some forms of reasoning can be reduced to routine steps that can be carried out by computers. But does this mean that a computer can have the "power of reason" in the same sense that we do? Can a computer have an intellect?

In thinking about this question, it is important to keep in mind that there is a distinction between being able to manipulate symbols correctly according to some prearranged scheme and understanding the meanings of those symbols. A very good illustration of this is provided by the simple little proof that we just gave of the formula $a + b + c = c + b + a$. What does this formula mean? Well, an obvious meaning is suggested by the conventional usage of the symbols

+ and =. Interpreted in this way, the formula is a statement about adding numbers. However, the formula could just as well be interpreted in other ways. For example, the symbol + could be taken to stand for multiplication, in which case the axioms, the rules of inference, and the entire proof are just as valid, as the reader can easily check. In fact, the symbols in the formula might have nothing to do with arithmetic at all. They might stand for words of the English language. The symbol = might stand for any form of the verb *to be*; a, b, and c might stand for nouns; and + might stand for a conjunction like *and*. In that case, the axioms, rules of inference, and proof are also valid. The formula $a + b + c = c + b + a$ could then mean that "Tom and Dick and Harry are Harry and Dick and Tom," or "bell and book and candle are candle and book and bell."

We see, then, that reducing a process of reasoning to the level of mechanical symbolic manipulation has the effect of draining it of most if not all of its meaning. What is left is form without specific content. That is why such manipulations are called "formal." When mathematicians reduce a branch of mathematics to such manipulations they say they are "formalizing" it, and the resulting system, with its symbols, rules for stringing symbols together into formulas, axioms, and rules of inference, is called a "formal system." Computers operate on this formal level, which is one reason that they cannot have any understanding of the meanings of the symbols they manipulate. A computer does not know whether the symbols it is printing out refer to numbers or to Tom, Dick, and Harry. It is therefore only in a very restricted sense that we can say that computers "reason." What they can do is the mechanical parts of reasoning, which involve no understanding of meaning and therefore do not require intellect.

When we speak of "understanding" we put ourselves in much the same position with respect to the materialist as when we speak of free will. For even though materialists, as much as anyone else, actually do understand things, including abstract concepts, they profess not to know what words like *understand* mean. In order to be admitted into their lexicon such words must be defined in what the materialist regards as a suitably "scientific" way. But as we do not yet have a theory that explains how we understand abstract concepts, and we cannot fashion probes that can be inserted into brains or computers to detect the presence of abstract understanding, it is hard to satisfy the demand put on us to define "scientifically" what we mean. Of course, this demand is unreasonable, since all scientific statements ultimately rely for their meaning (as Niels Bohr emphasized) on other statements made in everyday language whose meaning is derived from ordinary experience. One could not teach science to a person using only equations or technical vocabulary. Nevertheless, in dealing with materialists we tend to reach an impasse, since in speaking about phenomena, however much a part of ordinary experience, which are not easily or at all reducible to numbers, we are at risk of being accused of speaking of unreal things.

To make any headway, then, we must start not with the question of what computers can "understand," but with what they can do. That is, what can a computer compute? What can it give as output? Are there certain problems to which a computer is incapable of printing out an answer, but to which a human being can supply one? This is the kind of question that is involved in the argument put forward by the philosopher John Lucas, and later refined by the mathematician and physicist Roger Penrose, that aims to prove that human mental processes are not simply those of a computer. This argument takes as its starting point a powerful result in logic called Gödel's Theorem.[2] Gödel's Theorem, as originally formulated, referred not to computers but to "formal systems," that is, to branches of mathematics that had been completely reduced to symbolic language. However, there is a very intimate connection between formal systems and computer programs or algorithms. Both are essentially mechanical schemes of symbol manipulation. In fact, several years after Gödel proved his famous theorem about formal systems, similar theorems were proven about computers by Alan Turing and others. I shall therefore not bother to distinguish in the following discussions between formal systems and computer programs.

It is not hard to explain the gist of the Lucas-Penrose argument, as I already noted in chapter 3. In essence, Gödel showed that if one knew the program that a computer uses, one can in a certain sense outwit the computer. What Lucas and Penrose argued is that if we ourselves were just computers we would be able to know our own programs and thus *outwit ourselves*, which is clearly not possible. This is the Lucas-Penrose argument in a nutshell. In what follows, I shall give a more precise account of both of Gödel's Theorem and the Lucas-Penrose argument. In appendix C, I give a more mathematical account of how Gödel proved his theorem.

What Gödel Showed

The theorem proved in 1931 by the Austrian logician Kurt Gödel is about formal systems that have at least the rules of arithmetic and simple logic built into them. Such systems can be either "consistent" or "inconsistent." This is a very important distinction to understand, because Gödel's Theorem applies only to consistent formal systems. A system is called inconsistent if it is possible, using its rules, both to prove some proposition and to prove its contrary. For example, arithmetic would be inconsistent if we could prove both that $a = b$ and that $a \neq b$. (The symbol $\neq$ means "is not equal to.") A system is called consistent, on the other hand, if no such contradictions can arise in it.

There is a very important fact about inconsistent formal systems: if *any* contradictory statements can be proven, then *all* contradictory statements can be proven. This may be surprising, but it actually follows from the rules of elementary logic. Rather than explaining how one shows this in general, I will give an example. Suppose I set up the rules of arithmetic (badly) so that I can prove that 1 = 0. Using that one inconsistency, I can prove all arithmetic statements, whether they are true or false. For example, I can prove that 13 = 7. To do that, I just take 1 = 0, and multiply both sides by 6, to get 6 = 0. Then I just add 7 to both sides to get 13 = 7.

In other words, a formal system cannot be just a little inconsistent; if it is inconsistent, it is inconsistent all the way, thoroughly, radically, completely. In an inconsistent formal system *anything* that can be stated in that system can be proven using its rules.

There is an amusing story which illustrates this logical fact and which also shows how quick-witted the philosopher Bertrand Russell was. Russell, who did important work in logic as well as philosophy, was asked by an interviewer whether, given the fact that anything can be proven in inconsistent systems, he could use the statement "2 = 1" to prove that he (Russell) was the pope. Russell instantly proceeded to do just that. Consider (he said) a room containing 2 people, namely Bertrand Russell and the pope. Now, since 2 = 1, it is also true to say that there is only 1 person in the room. Since Russell is in the room, and the pope is in the room, and there is just 1 person in the room, then Russell and the pope must be the same person.

From the fact that *any* proposition which can be stated in an inconsistent formal system can be proven using its rules, a rather surprising conclusion follows: If we can find even one proposition that can be stated in a formal system but that *cannot* be proven using its rules, then we know that that system is completely consistent.

Having this basic distinction between consistent and inconsistent formal systems under our belts, we can say what it is that Gödel proved. What Gödel showed[3] is that in any consistent formal mathematical system in which one can do at least arithmetic and simple logic there are arithmetical statements which can neither be proved nor disproved *using the rules of that system* (i.e., using its axioms and rules of inference), but which nevertheless are in fact true statements. Statements that can neither be proved nor disproved using the rules of a formal system are called "formally undecidable propositions" of that system.

However, Gödel did much more than this. He also showed, for any particular consistent formal system containing logic and arithmetic, how to actually find one of its formally undecidable but true-in-fact propositions. The particular one he showed how to find is called the "Gödel proposition."

To sum up, if F is any consistent formal system that includes logic and arithmetic, then Gödel showed how to find a statement in arithmetic, which we may call G(F), that is *neither provable nor disprovable using the rules of* F; and he further showed that G(F) *is nevertheless a true arithmetical statement.*

What Gödel proved can be carried over to apply to computer programs. For a computer program P that is known to be consistent, and that is powerful enough to do arithmetic and simple logic, one can find a statement in arithmetic, G(P), that cannot be proven or disproven by that program. And one can show that G(P) is a true statement. It is in this sense that one has "outwitted" the computer, for one has succeeded in showing that a certain proposition is true that the computer itself cannot prove using its program.

Gödel proved one more thing in his famous theorem.[4] He showed that the consistency of a formal system is itself undecidable using the rules of that system. That is, if a formal system (or computer) is consistent, it cannot prove that it is.

The Arguments of Lucas and Penrose

In 1961, John R. Lucas, a philosopher at Oxford University, set forth an argument, based on Gödel's Theorem, to the effect that the human mind cannot be a computer program.[5] He wrote,

> Gödel's theorem seems to me to prove that Mechanism is false, that is, that minds cannot be explained as machines. So also has it seemed to many other people: almost every mathematical logician I have put the matter to has confessed to similar thoughts, but has felt reluctant to commit himself definitely until he could see the whole argument set out, with all objections fully stated and properly met. This I attempt to do.[6]

We shall explain the Lucas argument in a moment. First let me say a few words about its history. Gödel himself, though he did not lay out the details of an argument like Lucas's in public, seems himself to have reached the same conclusion. He did not believe that the human mind could be explained entirely in material terms. In fact, he called that idea "a prejudice of our times."[7] In this he was only too right; the prejudice that the mind is a computer has hardened in many minds to become a dogma. It was only to be expected, therefore, that Lucas's argument would be generally rejected when he advanced it in the 1960s, and that it would have little impact on the thinking of people who work in the field of artificial intelligence.

Recently, however, the eminent mathematician and mathematical physicist Roger Penrose (who also happens to be at Oxford) has revived Lucas's argument. His first book on the subject, *The Emperor's New Mind*, published in 1989,[8] provoked even more criticism than Lucas had. Much of this appeared in the journal *Behavioral and Brain Sciences* in 1990,[9] and Penrose answered it in a second book, *Shadows of the Mind*,[10] which also stimulated much debate.[11] One can best sum up the situation by saying that, while no one has succeeded in refuting the Lucas-Penrose argument, it has not succeeded in changing many minds. As we shall see, there are escape clauses in the argument that the materialist can make use of, but at the expense of diminishing the plausibility of his entire point of view. (It should be noted that Penrose, unlike Gödel and Lucas, seems to be a materialist himself, though an unusually open-minded one. He argues that while the human mind is not simply a computer it might still be explicable in physical terms in some as yet undreamt-of way.)[12]

Now I will explain the Lucas argument. First, imagine that someone shows me a computer program, P, that has built into it the ability to do simple arithmetic and logic. And imagine that I know this program to be consistent in its operations, and that I know all the rules by which it operates. Then, as proven by Gödel, I can find a statement in arithmetic that the program P cannot prove (or disprove) but which I, following Gödel's reasoning, can show to be a true statement of arithmetic. Call this statement G(P). This means that I have done something that that computer program cannot do. I can show that G(P) is a true statement, whereas the program P cannot do so using the rules built into it.

Now, so far, this is no big deal. A programmer could easily add a few things to the program—more axioms or more rules of inference—so that in its modified form it can prove G(P). (The easiest thing to do would be simply to add G(P) itself to the program as a new axiom.) Let us call the new and improved program P'. Now P' is able to prove the statement G(P), just as I can.

At this point, however, we are dealing with a new and different program, P', and not the old P. Consequently, assuming I know that P' is still a consistent program, I can find a Gödel proposition for *it*. That is, I can find a statement, which we may call G(P'), that the program P' can neither prove nor disprove, but which I can show to be a true statement of arithmetic. So, I am again ahead of the game.

However, again, this is no big deal. The programmer could add something to P' so that it too could prove the statement G(P'). By doing so he creates a newer and more improved program, which we may call P''. This race could be continued forever. I can keep "outwitting" the programs, but the programmer can just keep improving the programs. Neither I nor the programs will ever win. So, we have not proven anything. But here is where Lucas takes his brilliant step. Suppose, he says, that I myself *am* a computer program. That is, suppose that when I prove things there is just some computer program being run in my brain. Call that program H, for "human." And now suppose that I am shown *that* program. That is, suppose that I somehow learn in complete detail how H, the program that is *me*, is put together. Then, assuming that I know H to be a consistent program, I can construct a statement in arithmetic, call it G(H), that cannot be proven or disproven by H, but that I, using Gödel's reasoning, *can* show to be true.

But this means that we have been led to a blatant contradiction. It is impossible for H to be unable to prove a result that I am able to prove, because H is, by assumption, *me*. I cannot "outwit" myself in the sense of being able to prove something that I cannot prove. If we have been led to a contradiction, then somewhere we made a false assumption. So let us look at the assumptions. There were four: (*a*) I am a computer program, insofar as my doing of mathematics is concerned; (*b*) I know that program is consistent; (*c*) I can learn the structure of that program in complete detail; and (*d*) I have the ability to go through the steps of constructing the Gödel proposition of that program. If we can show that assumptions *b*, *c*, and *d* are valid, then we will have shown that *a* must be false. That is, we will have shown that I am not merely a computer program. This is Lucas's argument. Of course, there is nothing special about me. The same argument could be made about you or other human beings. But, in any event, as long as it applies to any human being, then that human being, at least, is not a mere computer.

From the foregoing, we see that Lucas's argument has possible loopholes, or avenues of escape, that can be used by those pitiable people who cling to the belief that they are computers. They can, instead of denying *a*, deny the assumptions *b*, *c*, or *d* of the proof. I have given here the argument as Lucas originally gave it. Roger Penrose presents it in a version that is slightly different and in his view somewhat stronger. He has also given answers to the numerous objections that have been raised against it.

Avenues of Escape

I cannot possibly discuss here every version of every objection that has been raised to the Lucas-Penrose argument or every way of attempting to escape its conclusions. I will, however, try to make clear what the main issues are.

Let us start with the avenue of escape for assumption *c*: that a human being cannot know his own computer program. As a practical matter, this is undoubtedly true. In the first place, to understand the structure of any particular human brain in complete detail would almost certainly entail procedures that would be so invasive as to destroy the brain in question. Thus, it is hard to see how a person could know his *own* brain's program. Nevertheless, a person might be able to discover the program of someone else's brain, and it might be argued that all human brains work in basically the same way; but this last point can certainly be disputed.

However, I think there is a much better answer. If our thinking is really just the running of a computer program, it does not matter for the Lucas argument what kind of machine that program is run on. The same program can run on a computer that uses vacuum tubes, or silicon chips, or neurons. One can certainly imagine a world in which the very same program that my brain uses is run on a kind of machine whose internal parts and programming are easily open to self-inspection. That machine (according to the materialist viewpoint) ought to be able to prove the same things that I can prove. But that machine, unlike myself, could know its own structure without any invasive or destructive procedures. The Lucas-Penrose argument can be applied to that machine to reach the same contradiction as before, but without escape avenue *c*.

Now let us turn to the escape avenue for assumption *d*. Suppose that I am able to have access to complete information on the structure of my brain's program. Is it not possible that the quantity and the complexity of that information is so great that I would die of old age before I was able to analyze it and construct the appropriate Gödel proposition for it? Again, this objection is based upon the details of how my body and brain are constructed, and in particular upon the fact that I will die of old age. There is no reason, however, that my program could not, in principle, be run on a machine that was far more durable and reliable in an engineering sense than I am.

A more significant objection is based on the possibility that the Gödel proposition of the human program, G(H), is so complex that the human mind simply does not have the capacity to construct it. This is another version of escape avenue *d*. This objection is based on a possible limitation of the human program itself rather than of the human machine that runs it. The idea that the finiteness of the human mind might prevent it from analyzing its own program in the way supposed in the Lucas-Penrose argument has some plausibility. The issue has to do with how complicated G(H) is. This is a quantitative question, and can therefore be analyzed mathematically. Penrose has done this and concludes that the degree of complexity of G(H) would be virtually the same as that of H itself, so that if a brain can learn what its own program is, it can probably also do the computations necessary to construct its Gödel proposition.[13]

Aside from the technicalities of Penrose's analysis of this question, there is a more basic consideration. How likely is it that natural selection could have produced a program so complicated that it defies human analysis even given an unlimited amount of patience and time? (Recall that we can always do the Lucas-Penrose argument assuming that the human program is run on a machine that does not wear out.) Our forebears are supposed to have diverged from those of chimpanzees about 6 to 8 million years ago. So within a few hundred thousand generations humans developed the ability to do abstract mathematical reasoning. In fact, that breakthrough certainly happened much more recently. The genus *Homo* has only been around for about 2 million years, and *Homo sapiens* has only been around for a few hundred thousand years, or about ten thousand generations.) Moreover, from generation to generation the changes in the structure of the brain and its program were presumably fairly slight, and these changes

were governed by a haphazard process of trial and error. It seems highly implausible, therefore, that a physical structure constructed in this blundering way in a limited period of time could not, even in principle, yield to an intelligent analysis which had unlimited time at its disposal. (Again, we are assuming access to complete information about the actual physical structure of the brain.)

We come now to the escape avenue for assumption *b*. The most popular objection to the Lucas-Penrose argument seems to be that human beings are inconsistent computer programs, or at least do not know themselves to be consistent. At first glance this seems incontestable. After all, who is it that has never reasoned inconsistently or made mistakes in mathematics?

In response to this objection two points must be made. The first is that there is a difference between a computer that is using a truly inconsistent program and one that, though using a consistent program, malfunctions because of interference with its normal scheme of operation. A computer which has a perfectly consistent program for doing arithmetic may nevertheless give a wrong answer if, for example, a cosmic ray switches a bit from "0" to "1" in its memory, or if some circuit element or chip develops a physical flaw, due perhaps to excessive heat. It seems plausible to suppose that many human errors, such as those that arise from fatigue or inattention, are due to malfunctioning.

The second point is that if we are programs that are really, in themselves, inconsistent then we must be radically inconsistent, as I explained before. A truly inconsistent program or formal system can prove *anything whatever*. Indeed, it is just for that reason that it evades the Lucas-Penrose argument. That argument was based upon the inability of H to prove G(H); but if H is inconsistent it can prove *anything*. Thus, an inconsistent program is vastly more powerful than a consistent one, just as a liar is able to assert more things than an honest person can. But while more powerful, in a sense, an inconsistent program is also quite helpless, for *it cannot recognize its own mistakes*. An inconsistent program would be just as able to prove that 2 + 2 = 17 as that 2 + 2 = 4, and would not have any way to tell which answer to prefer. It would, therefore, be equally satisfied with either result. It is quite otherwise with humans. We do make mistakes in our sums, but we can also spot those mistakes, or have them pointed out to us, and recognize that they *are* mistakes. As Lucas observed,

> If we are really inconsistent machines, we should remain content with our inconsistencies, and happily affirm both halves of a contradiction. Moreover, we would be prepared to say absolutely anything—which we are not. . . . This surely is a characteristic of the mental operations of human beings: they are selective; they do discriminate between favoured—true—and unfavoured—false—statements; when a person is prepared to say anything, and is prepared to contradict himself without any qualm or repugnance, then he is adjudged to have "lost his mind." Human beings, although not perfectly consistent, are

not so much inconsistent as fallible. A fallible but self-correcting machine would still be subject to Gödel's results. Only a fundamentally inconsistent machine would escape.[14]

Again, it must be emphasized that an inconsistent system has no way to resolve contradictions. In such a system $2 + 2 = 4$ is in no way better than $2 + 2 = 17$ or any other result. It is possible that an inconsistent machine could be prevented from ever actually making explicitly contradictory statements: it could have a "stop order," so that if it ever said or printed out "$2 + 2 = 5$" it would prevent itself from later saying or printing out "$2 + 2 \neq 5$." Such a machine could never be "caught" in a contradiction, but would still be radically inconsistent in a way that humans are not. Again to quote Lucas:

> No credit accrues to a man who, clever enough to see a few moves of argument ahead, avoids being brought to acknowledge his own inconsistency, by stonewalling as soon as he sees where the argument will end. Rather, we account him inconsistent too, not, in his case, because he affirmed and denied the same proposition, but because he used and refused to use the same rule of inference. A stop-rule on actually enunciating an inconsistency is not enough to save an inconsistent machine from being called inconsistent."[15]

Humans, therefore, have an ability to reason consistently in a way that an inconsistent machine would not have and would not be able to fake.

There is another possibility, and that is that the human program, H, has built into it a rule which is inconsistent, but which only comes into play in some very esoteric situations which rarely come up, or perhaps have never yet come up in human experience. A mathematician, for example, might be happily going along thinking that $1 \neq 0$, when suddenly he would be confronted with an argument that called into play that inconsistent rule that was lurking in his brain's program, and this argument would convince him that $1 = 0$ after all. Nor would he be able to find his mistake, for by the rules followed by his mental processes—the rules that define what seems "rational" to him—there would be no mistake. The argument that proved that $1 = 0$ would always seem quite reasonable to him. Indeed, because he is inconsistent, *any* proposition would be provable to him by some argument that he would find perfectly valid.

If human beings were like this, then we would obviously be utterly irrational in the final analysis. All human reasoning would be an exercise in futility. Here we get down to the basic issue: Are we rational, and do we know that we are?

It is not a question whether human beings are sometimes irrational. As Lucas observed, there is a difference between fallibility and radical inconsistency. The question is whether we can *ever* be truly rational and *know* that we are being rational. Is there ever a time when I can be sure that a particular statement is true or piece of reasoning is valid and that the contrary is not? I assert that there

is. For instance, I claim to know that 1 is not equal to 0. Moreover, I also claim to know that no argument that I would recognize as a valid one would ever lead me to conclude that 1 is equal to 0. Or, rather, if I ever did fall for some fallacy that implied that $1 = 0$, I would at least be able to recognize it as a fallacy. This I claim to know about myself, and I believe that any reasonable person knows this about himself too. And if I know this, then I know that I am not an inconsistent program.

What, then, do I say to someone who points out that I have often claimed to know something in the past only to have it shown that I was in error? If my certainty was illusory in the past, then is it not possible that all my certainties may be illusory? Not necessarily. We often speak rather loosely about certainty. In many cases we mean simply a practical or "moral" certainty, as in our certainty that the Sun will come up tomorrow, not a real certainty as in a certainty that $1 \neq 0$. In addition, there are times when we are delusional. When asleep, for example, we sometimes dream that we are awake. In fact, in our dream we may even feel quite confident that we are awake. Notwithstanding this, I think few people would deny that there are times when we really do know that we are awake, and know that we are not deluded in thinking so. This is a paradoxical fact, perhaps, but a fact nonetheless.

One of the amusing aspects of the debate about the Lucas-Penrose argument is that many of those who claim that human beings are inconsistent programs, and who therefore are committed to the view that genuine certainty is impossible, nevertheless act as though they are quite certain of one thing—namely that the Lucas-Penrose argument is wrong!

It is very interesting that one of the most common objections made against the Lucas-Penrose argument involves the claim that human beings are fundamentally inconsistent. It is strange that in making this objection many materialists feel that they are fighting the good fight against what they regard as the superstitious idea of a "soul." It used to be that those who rejected religious tenets usually did so in the name of human reason. They called themselves "rationalists." But the new kind of skepticism is willing, in order to debunk the spiritual in man, to call into question human reason itself. According to this view, we are not even supposed to be able to trust ourselves about the simplest truths of arithmetic. G. K. Chesterton, with prophetic insight, saw where things were heading almost a century ago:

> Huxley preached a humility that is content to learn from Nature. But the new sceptic is so humble that he doubts if he can even learn. . . . We are on the road to producing a race of men too mentally modest to believe in the multiplication table. We are in danger of seeing philosophers who doubt the law of gravity as being a mere fancy of their own. Scoffers of old were too proud to be convinced; but these are too humble to be convinced.[16]