Data Access Corporation v Powerflex Services Pty Ltd [1999] HCA 49

HIGH COURT OF AUSTRALIA

GLEESON CJ, GAUDRON, McHUGH, GUMMOW AND HAYNE JJ

GLEESON CJ, McHUGH, GUMMOW AND HAYNE JJ:

1 Pursuant to the grant of special leave to appeal, Data Access Corporation ("Data Access") appeals to this Court against orders[1] of the Full Court of the Federal Court (Black CJ, Hill and Sundberg JJ) which were based on a holding that copyright as original literary works did not subsist in commands in the Dataflex computer language contained in a computer program developed by Data Access in the United States. The Full Court, reversing the judgment of the trial judge, Jenkinson J[2], held that those commands were not "computer programs" within the meaning of the definition in s 10 of the *Copyright Act* 1968 (Cth) ("the Act") and were not entitled to protection under that statute.

2 The provisions of the Act apply to the proceedings because, although Dataflex appears first to have been published in the United States, the international arrangements implemented by s 184(1)(a) of the Act and by reg 4(1) of the Copyright (International Protection) Regulations (Cth) treat Dataflex as if it had been first published in Australia[3].

3 Also involved in the proceedings in this Court is a cross-appeal in which the respondents challenged the finding by the Full Court of the Federal Court that they had infringed the copyright in a data table used by the Dataflex program in compressing data, known as the "Huffman compression table". Special leave has not been granted in respect of this issue, but the parties have argued the issues on the cross-appeal on the basis that the special leave application and the substantive appeal should be heard together.

---

1 *Powerflex Services Pty Ltd v Data Access Corporation (No.2)* (1997) 75 FCR 108 at 132.

2 *Data Access Corporation v Powerflex Services Pty Ltd* (1996) 63 FCR 336.

3 *Data Access Corporation v Powerflex Services Pty Ltd* (1996) 63 FCR 336 at 337.

The factual background

4 Data Access owns the copyright in a system of computer programs which is known as "Dataflex". The relationship between a computer and a program is described by Carr and Arnold[4] as follows:

> "A program is executed by the central processing unit (CPU) of the computer, which is the centre of control for arithmetical and logic operations within the microprocessor. The CPU consists of an arrangement of electronic circuits which are activated by impulses of electric current. A logic gate within the CPU is either turned on or off depending on the presence or absence of such pulses.
>
> The presence or absence of pulses of current is represented by binary digits ('bits'). The CPU recognises '1' as indicative of the presence of a pulse, and '0' as indicative of its absence.
>
> A computer program is a series of bits, each bit representing the presence or absence of a pulse. The program operates within the CPU as a series of pulses in a prearranged sequence in accordance with the order of bits devised by the computer programmer. Accordingly, the 'instructions' of a computer program represent a series of impulses which operate within the computer to make the machine perform certain predefined functions.
>
> Each instruction is held in a memory location within the computer, which has an address, by which the location may be identified. The addresses of the memory locations in which program instructions are to be found are held in the program counter. The CPU locates the address of each instruction of the program, accesses it from the memory location, from where it is reproduced and executed in the instructions register. The program counter may contain a 'jump' instruction, which makes the CPU jump beyond the next instruction in sequential order in the software, to access and execute a subroutine."

5 In *Computer Edge Pty Ltd v Apple Computer Inc*[5], Gibbs CJ gave a more detailed account of the general nature of a computer program and how it causes a computer to function. Substantial developments and improvements to computers and computer technology have occurred since the trial of that action in 1983[6]. For example, evidence in that case appears to have shown that the first stages of

---

4 Carr and Arnold, *Computer Software: Legal Protection in the United Kingdom*, 2nd ed (1992) at 1-2.

5 (1986) 161 CLR 171 at 178-179.

6 *Apple Computer Inc v Computer Edge Pty Ltd* (1983) 50 ALR 581.

a construction of the programs in 1977 were handwritten[7], whereas many programs are now written entirely on computer[8]. Nevertheless, much of what was said by Gibbs CJ remains relevant. His Honour said[9]:

> "Computer science makes much use of jargon and metaphor, and to enable the matters in issue to be understood it seems desirable to attempt a brief explanation of the meaning of some of the expressions used in that science and to describe the manner in which a computer program is developed. A computer program is a set of instructions designed to cause a computer to perform a particular function or to produce a particular result. A program is usually developed in a number of stages. First, the sequence of operations which the computer will be required to perform is commonly written out in ordinary language, with the help, if necessary, of mathematical formulae and of a flow chart and diagram representing the procedure. In the present case if any writing in ordinary language (other than the comments and labels mentioned below) was produced in the production of Applesoft and Autostart, no question now arises concerning it. Next there is prepared what is called a source program. The instructions are now expressed in a computer language – either in a source code (which is not far removed from ordinary language, and is hence called a high level language) or in an assembly code (a low level language, which is further removed from ordinary language than a source code), or successively in both. Sometimes the expression source code seems to be used to include both high level and low level language. In the present case, the source programs were written in an assembly code, comprising four elements, viz. – (a) labels identifying particular parts of the program; (b) mnemonics each consisting of three letters of the alphabet and corresponding to a particular operation expressed in 6502 Assembly Code (the code used); (c) mnemonics identifying the register in the micro-processor and/or the number of instructions in the program to which the operation referred to in (b) related; and (d) comments intended to explain the function of the particular part of the program for the benefit of a human reader of the program. The writing has been destroyed, although it is possible to reconstruct the mnemonics, but not the labels and comments, which were comprised in it.
>
> The source code or assembly code cannot be used directly in the computer, and must be converted into an object code, which is 'machine readable', i.e., which can be directly used in the computer. The conversion

---

7 (1983) 50 ALR 581 at 587-590.

8 Cornish, *Intellectual Property: Patents, Copyright, Trade Marks and Allied Rights*, 3rd ed (1996) at 442-443.

9 (1986) 161 CLR 171 at 178-179.

is effected by a computer, itself properly programmed. The program in object code, the object program, in the first instance consists of a sequence of electrical impulses which are often first stored on a magnetic disc or tape, and which may be stored permanently in a ROM ('read only memory'), a silicon chip which contains thousands of connected electrical circuits. The object code is embodied in the ROM in such a way that when the ROM is installed in the computer and electrical power is applied, there is generated the sequence of electrical impulses which cause the computer to take the action which the program is designed to achieve. The pattern of the circuits in the ROM may possibly be discerned with the aid of an electron microscope but it cannot be seen by the naked eye. Obviously, the electrical impulses themselves cannot be perceived. However the sequence of electrical impulses may be described either in binary notation (using the symbols 0 and 1) or in hexadecimal notation (using the numbers 0-9 and the letters A-F), and it is possible to display the description on the visual display unit of the computer, and to print it out on paper. And, as has been said, it is also possible to reconstruct the mnemonics in the source code. It will have been seen from this account that a program exists successively in source code and in object code, but the object code need not be written out in binary or hexadecimal notation in the process of producing and storing the program."

6 In the judgments in *Computer Edge* of Gibbs CJ[10] and the other members of the Court[11], reference is made only to the *entirety* of the object code of "Applesoft" and "Autostart". The litigation dealt with the Act before its amendment by the *Copyright Amendment Act* 1984 (Cth), s 3 of which introduced the definition of "computer program" into s 10(1). The amendment also brought a "computer program" within the ambit of a "literary work". In *Computer Edge*, no point was taken that less than the entirety of the routines involved might constitute a program and qualify as an original literary work. In the present case, however, the appellant claims that individual words can constitute a "computer program" for the purposes of s 10(1) of the Act.

The Dataflex system

7 Development of the Dataflex application development system commenced in 1979, and it was first published in 1981. Dataflex has been the subject of 18 revisions since that time, is well known, and has achieved a significant share of the market for programs of its kind. It is a system which allows a programmer or developer to develop customised database applications or databases.

---

10 (1986) 161 CLR 171 at 179.

11 (1986) 161 CLR 171 at 190-191 per Mason and Wilson JJ; 198-200 per Brennan J; 210-211 per Deane J.

8 The Dataflex system incorporates:

- a program development system, which provides the means to write, edit, compile and run programs under development;

- a computer programming language known as the "Dataflex language", which is an application development language in which the source code for all Dataflex programs is written or generated;

- a compiler program which translates the Dataflex source code programs written in the Dataflex language by using the program development system, into an internal format which is then able to be run by the runtime program; and

- a runtime program, which is the executable program required to run the compiled application programs developed using the program development system.

9 Mr Cory Casanave, Executive Vice-President of Data Access, gave evidence as to the nature of the language used in computer programs. He said:

> "A computer language defines the names of each word in the language and the rules governing the use of each word (syntax). Each word in a computer language is an instruction to the computer to invoke lower level processes, the word chosen to invoke those processes is generally chosen to suggest the nature of the process that will be invoked.
>
> A computer language is comprised of a set of reserved words which are used in accordance with the rules of syntax governing their use. A computer language syntax, like the syntax of a human language, comprises the rules by which the words can be combined to form statements which are correct for the language. For each command or function there is a specific syntax which describes how arguments may be applied to the command. Arguments can be likened to a noun phrase, they describe what the command will act on. Various documents also refer to 'functions' as well as commands. Functions are a type of command which perform a computation and return a result."

10 Two hundred and fifty-four words of the Dataflex language are listed in the Dataflex encyclopedia. However, 29 of those words, which express commands for developing graphics, are not used at all in the PFXplus language. Of the other 225, 192 are used in the PFXplus language in a way which eventually causes the computer to perform the same function as those words perform in the Dataflex language.

11 At first instance, the reasons of Jenkinson J suggest that he considered that copyright subsisted in each of these 192 common words and that the use of them

in PFXplus infringed Data Access' copyright in them[12]. However, the orders made by his Honour restrain the use of only 169 of the common words, three of which are "Macros". It may be that, after judgment but before the orders were made, Data Access conceded that the remaining 23 words (which include words such as "SHOW" and "ENTRY") were not protected by copyright. Perhaps a concession was made during argument and, in a complex case, overlooked when the reasons for judgment were being prepared. But whatever the reason for not dealing with these 23 words, the issue of copyright in this Court is confined to the 169 words the subject of the order made by his Honour.

12 We will deal with the three "Macros" separately from the remaining 166 common words which can conveniently be called the "Reserved Words". Of these Reserved Words, at least 55 are unique to the Dataflex program. But many are ordinary English words – such as "BOX", "CHART", and "RETAIN". Others are a combination of two English words such as "PAGEBREAK". Some are not only common English words but are used in most, if not all, computer programs. Examples are "DIRECTORY" and "SAVE".

The PFXplus system

13 Some years ago, the third respondent, Dr David Bennett ("Dr Bennett"), became familiar with the Dataflex system. He decided to create and market an application development system which would be compatible with the Dataflex language and the Dataflex database file structure so that persons who were familiar with the Dataflex system would be able to use his new product.

14 The Full Court of the Federal Court found the following facts to be common ground between the parties to the appeal:

– by a process of reverse engineering and study of both the documentation and operation of the Dataflex system, Dr Bennett created a system of computer programs, which was originally known as "Powerflex", but is now known as "PFXplus", intending that the system would be compatible with the Dataflex system, i.e. that certain commands and "reserved words" (including the Macros) used as commands in the Dataflex system would operate in like manner in the PFXplus system;

– the source code in which the Dataflex system is written is quite different from the source code in which the PFXplus system is written; and

– there is not necessarily any similarity between the object code used in the Dataflex system and that used in the PFXplus system.

---

12 *Data Access Corporation v Powerflex Services Pty Ltd* (1996) 63 FCR 336 at 339.

15 PFXplus achieved Dr Bennett's aim of being highly compatible with Dataflex. Dr Bennett and his wife subsequently incorporated a company, the second respondent, to sell PFXplus.

The issues

16 The only allegations of copyright infringement that are now in issue are the claims that by publishing PFXplus the respondents have infringed the copyright which Data Access has in:

A. The Reserved Words.
B. The Macros.
C. The Dataflex Huffman compression table, to which reference will later be made.

A. THE RESERVED WORDS

17 The appellant contends:

1. Copyright subsists in each of the Reserved Words because each is a "computer program" within the definition in s 10(1) of the Act.

2. Copyright subsists in the collocation of the Reserved Words comprising the Dataflex language because this collocation is a "computer program".

3. Alternatively to 2, even if the collocation of Reserved Words is not itself a literary work, it nevertheless forms a substantial part of a literary work (the Dataflex system) so that copying of it is an infringement of the appellant's copyright in the Dataflex system.

4. Copyright subsists in the table or compilation of Reserved Words in the Dataflex User's Guide on the footing that it is a literary work within par (a) of the definition in s 10(1) of the Act.

We turn to consider each of these contentions.

1. *Is each of the Reserved Words a "computer program" within the meaning of s 10(1) of the Act?*

18 The appellant contends that each of the Reserved Words is itself a "computer program" within the meaning of the definition in s 10(1) of the Act. In order to determine the validity of the appellant's submissions, it is convenient to divide the definition of "computer program" into its component parts.

19 The definition of "computer program" requires that each Reserved Word be:

(i) "an expression,"
(ii) "in any language, code or notation,"
(iii) "of a set of instructions (whether with or without related information)"
(iv) "intended, either directly or after either or both of the following:
(a) conversion to another language, code or notation;
(b) reproduction in a different material form;
to cause"
(v) "a device having digital information processing capabilities to perform a particular function."

20 Each of the first four of these elements qualifies what follows and the scope of the definition is marked out by the requirement of an intention that the device be caused "to perform a particular function". In form, the definition of a computer program seems to have more in common with the subject matter of a patent than a copyright. Inventions when formulated as a manner of new manufacture traditionally fell within the province of patent law, with the scope of the monopoly protection being fixed by the terms of a public document, the patent specification. In Australia claims to computer programs which are novel, not obvious and otherwise satisfy the *Patents Act* 1990 (Cth) and which have the effect of controlling computers to operate in a particular way, have been held to be proper subject matter for letters patent, as "achieving an end result which is an artificially created state of affairs of utility in the field of economic endeavour"[13], within the meaning of *National Research Development Corporation v Commissioner of Patents*[14].

21 The amendment of the definition of "literary work" in s 10(1) of the Act to include as item (b) "a computer program or compilation of computer programs" obviously marked a significant departure from what previously had been the understanding of what was required for subsistence of copyright in an original literary work. It is true that copyright may subsist in a literary work which is related to the exercise of mechanical functions. A set of written instructions for the assembly and operation of a domestic appliance is an example. However, it is not to the point in copyright law that, if followed, the instructions do not cause the appliance to function. The protection of the function performed by the appliance will be for the patent law, including the law as to inutility. This is what was indicated by Bradley J in a passage in *Baker v Selden*[15] which was repeated by Brennan J in *Computer Edge*[16]. Bradley J said that no one would contend that the exclusive right to the manner of manufacture described in a

---

13 *CCOM Pty Ltd v Jiejing Pty Ltd* (1994) 51 FCR 260 at 295.

14 (1959) 102 CLR 252 at 275-277.

15 101 US 99 at 102 (1879).

16 (1986) 161 CLR 171 at 208-209.

treatise would be given by the subsistence of copyright in that work, and continued:

> "The copyright of the book, if not pirated from other works, would be valid without regard to the novelty, or want of novelty, of its subject-matter. ... To give to the author of the book an exclusive property in the art described therein, when no examination of its novelty has ever been officially made, would be a surprise and a fraud upon the public. That is the province of letters-patent, not of copyright."

22 Further, the requirement in copyright law that a work be "original" is to be distinguished from the requirements that an alleged invention be novel and that it not be obvious[17]. The question for copyright law is whether "the work emanates from the person claiming to be its author, in the sense that he has originated it or brought it into existence and has not copied it from another"[18]. If so, the work does not lack originality because of the anterior independent work of another, although, in such circumstances, an invention might lack novelty.

23 Finally, to say that the copyright law does not protect function and extends only to the expression of systems or methods[19] does not deny that a work may serve utilitarian rather than aesthetic ends. A map and a recipe book are obvious examples.

24 There is, with respect, some oversimplification of these principles in the following statement by Dawson J in *Autodesk Inc v Dyason*[20] ("*Autodesk No 1*"):

> "[W]hen the expression of an idea is inseparable from its function, it forms part of the idea and is not entitled to the protection of copyright".

25 The 1984 amendment departed from traditional principles by identifying for copyright purposes a species of literary work, the very subsistence of which requires an expression of a set of instructions intended to cause a device to perform a particular function. The difficulties which arise from accommodating computer technology protection to principles of copyright law have been remarked upon[21] but the Act now expressly requires such an accommodation.

---

17 *Sands & McDougall Pty Ltd v Robinson* (1917) 23 CLR 49 at 53.

18 Ricketson, *The Law of Intellectual Property*, (1984) at 83.

19 *Hollinrake v Truswell* [1894] 3 Ch 420 at 427, 428.

20 (1992) 173 CLR 330 at 345.

21 Cornish, *Intellectual Property: Patents, Copyright, Trade Marks and Allied Rights*, 3rd ed (1996) at 181-182; Karjala, "A Coherent Theory for the Copyright

(Footnote continues on next page)

26 In the present case, no question arises with respect to the relationship between the Act and the protection given to the designers of computer chips by the *Circuit Layouts Act* 1989 (Cth)[22]. The first issue in the appeal turns solely on the application of the definition of "computer program" in s 10(1) of the Act.

27 The appellant submits that each Reserved Word meets each component of the definition of "computer program"[23]. The appellant contends:

(i) A Reserved Word itself is identified as being the relevant "expression" for the purposes of the definition. The term "expression" is used in the definition to preserve the distinction between the set of instructions and the manner in which the set is expressed in a particular programming language. The choice of expressions for words and commands in a language is determined by the author of the language.

(ii) Each Reserved Word is in a code or notation, the relevant code or notation being the Dataflex language.

(iii) Each Reserved Word expresses a set of instructions, that set being either the underlying set of instructions in source code, or the meaning and syntax of the word or command in question.

(iv) and (v) Each Reserved Word is in a high level language, and each is intended, after conversion into a lower level language by a compiler and runtime program, to cause a computer (which is a device having digital information processing capabilities) to perform a particular function.

28 In our opinion, none of the Reserved Words satisfies the statutory definition. Each Reserved Word is undoubtedly in "code or notation" – the Dataflex language. It follows that whether a Reserved Word is a "computer program" within the meaning of the definition depends on whether it is an "expression ... of a set of instructions ... intended ... to cause a device having digital information processing capabilities to perform a particular function". However, each of the Reserved Words is a single word; none is a set of instructions in the Dataflex language. Further, none of the Reserved Words intends to express, directly or indirectly, an algorithmic or logical relationship

---

Protection of Computer Software and Recent Judicial Interpretations", (1997) 66 *University of Cincinnati Law Review* 53 at 56-66.

22 See *Avel Pty Ltd v Wells* (1992) 36 FCR 340 at 343-346; *Nintendo Co Ltd v Centronics Systems Pty Ltd* (1994) 181 CLR 134 at 141-143.

23 See par [19] above.

between the function desired to be performed and the physical capabilities of the "device having digital information processing capabilities".

29 We turn to explain these conclusions.

The findings in the courts below

30 The trial judge held that each Reserved Word was itself a computer program. Jenkinson J said[24]:

> "Each of the words of the DataFlex language found also in the PFXplus language is in my opinion an expression of a set of instructions intended to cause a device having digital information processing capabilities to perform a particular function. The circumstance that the expression of those instructions in source code is different is in my opinion immaterial. At the level of abstraction under consideration the objective similarity is complete: the set of instructions intended to cause the performance of the particular function is expressed, at that level where the 'language, code or notation' is based upon concatenations of letters of the alphabet, by the same concatenation of letters in each language. If at that level some of the concatenations constitute or resemble words of the English language descriptive or suggestive of the functions to be performed, that may facilitate the use of the computer program by those who understand English. But each concatenation of letters is nonetheless an expression of a set of instructions intended to cause the device to perform a particular function, in my opinion, and therefore a 'computer program' within the meaning of that expression in the *Copyright Act*."

31 The Full Court came to the opposite conclusion. It said[25]:

> "Each of the words in the so-called Dataflex language is but a cipher. The underlying program is the set of instructions which directs the computer what to do when that cipher is in fact used, for example by being typed on to the screen. It is not to the point that the cipher bears some resemblance to an ordinary English word. The cipher or command is not an expression of the set of instructions, although it appears in that set of instructions. It is the trigger for the set of instructions to be given effect to by the computer.
>
> It may not be inaccurate to describe each of the commands as itself an instruction. It is likewise not necessarily inaccurate to talk of each of those words as representing the set of instructions in the sense that the use of one

---

24 (1996) 63 FCR 336 at 339.

25 (1997) 75 FCR 108 at 122.

of them triggers the instructions contained in the computer program to be acted upon. But it is in our view not accurate to refer to each of the words as being an expression of the set of instructions. The set of instructions is expressed in the source code which is the computer program and, at least at a higher level, includes the particular word which is a command. The computer program will also in other forms exist in lower level language, ultimately through to an object code in non-visible form. Each of these representations will fall within the definition of 'computer program'. In each of them, in some language, code or notation, the word said to be part of the computer language will be able to be found.

The passage earlier quoted from the judgment of Gaudron J in *Autodesk,* placing as it does emphasis upon the requirement that it is the set of instructions in their entirety which is the computer program, also points to the conclusion that the individual words of command are not, themselves, computer programs within the definition."

Previous judicial consideration of the term "set of instructions"

32 The passage in Gaudron J's judgment in *Autodesk* referred to by the Full Court is found in *Autodesk Inc v Dyason [No 2]*[26] ("*Autodesk No 2*"), where, in discussing the definition of "computer program" in s 10(1) of the Act, her Honour said[27]:

"[I]t is, in my view, clear that that expression directs attention to an entire instruction or, more accurately, an entire set of instructions, and not merely those parts that consist of bare commands. So much is confirmed by the language used in the definition and by its context. The words 'set of instructions' necessarily direct attention to instructions in their entirety. And that direction is in no way cut down, but, rather, is reinforced by the parenthetical description of the instructions involved as instructions 'whether with or without related information'. Moreover, the definition is concerned with instructions which 'cause a device having digital information processing capabilities to perform a particular function' and in many cases it will be necessary for instructions to be accompanied by related information if those devices are to perform quite ordinary computer functions."

---

26 (1993) 176 CLR 300.

27 (1993) 176 CLR 300 at 329.

33 In *Autodesk No 2* Mason CJ, in a passage[28] which was also quoted with evident approval by the Full Court[29], said that a fundamental principle confirmed by *Autodesk No 1*[30] was that:

> "the definition of a 'computer program' by reference to 'an expression ... of a set of instructions' should be understood as conferring protection upon the set of instructions itself – which must be identified with some precision – but as doing so in a way which is adapted to the nature of copyright. Thus, the protection of computer programs is to conform to the dominant principle of copyright law that protection is given not for ideas, but only for the form of expression. However, as the judgment of Mason CJ, Brennan and Deane JJ makes clear, this distinction must not be applied too strictly. A distinction needs to be drawn between the relevant set of instructions and the form of storage or representation of the instructions, so that a person who reproduces a set of instructions in a different form – such as by turning source code into object code – does not escape infringement. The object of protection is the computer program, not just the particular form of storage or representation chosen by the author." (footnotes omitted)

This passage indicates that it is necessary to identify the "set of instructions" with some precision.

34 In construing the expression "set of instructions" in the definition of "computer program" in s 10(1), there is no need to choose between the "technical" meaning and the "natural and ordinary" meaning of the expression[31]. Words in a statutory definition are to be interpreted in their statutory context. A court cannot interpret the meaning of the definition of "computer program" in s 10(1) without some understanding of the manner in which a computer executes computer programs. If interpreting the phrase "set of instructions" in the definition of "computer program" in light of the manner in which a computer operates is to be regarded as ascribing a technical meaning to the phrase, then in our opinion it must be given its technical meaning. However, in our opinion the "natural and ordinary" meaning and the "technical or art" meaning of the phrase "set of instructions" is one and the same when it is considered in its statutory context.

---

28 (1993) 176 CLR 300 at 303-304.

29 (1997) 75 FCR 108 at 119.

30 (1992) 173 CLR 330.

31 See *Collector of Customs v Agfa-Gevaert Ltd* (1996) 186 CLR 389 at 401-402; cf *Autodesk No 2* (1993) 176 CLR 300 at 329-330.

The relevant "set of instructions" executed by a computer

35 As the Full Court pointed out[32]:

> "A computer system consists of hardware and software. The hardware includes a central processing unit which contains the electronic circuits which control the computer and perform the relatively simple arithmetical calculations and logical operations (ie comparing values to determine which is larger) of which the computer is capable. It is because the computer is able to perform millions of such operations each second that computers may be used to perform extremely complex calculations and functions."

36 It is impossible to overemphasise the importance of the fact that a computer has no "intelligence" to execute instructions over and beyond the simple logical functions which are hard wired into its circuits[33]. In order for the simple logical functions of a computer to translate into a useful result, it is necessary to express complex problems in terms of a sequence of a large number of these simple operations. A "set of instructions" will not cause a computer to execute a particular function unless that set of instructions can be ultimately expressed in terms of a sequence of the logical operations which are hard wired into the computer. No doubt it is very rare to express a complex computer program in terms of the simple logical operations which are hard wired into a computer. That is because the process of writing programs becomes practically unmanageable unless the "set of instructions" is perceived at a high level of abstraction. Such a level of abstraction is required in order to express what are millions of simple logical operations in terms of a manageable number of more complex instructions which themselves are reducible to these simple logical operations.

37 An example of, and a commentary upon, the varying levels of abstraction at which a set of instructions can be viewed is given by Mr Prescott QC in "Copyright and microcomputers – Some current legal problems"[34]. Suppose we wished to write a computer program in a high level language such as BASIC to cause a "device having digital information processing capabilities" to perform the "particular function" of printing the character "*" in the middle position on the screen. The relevant set of instructions could be stated shortly and clearly in BASIC as follows:

```
10 PRINT @ 544 '*'
```

---

32 (1997) 75 FCR 108 at 111.

33 For a discussion of these principles see Laddie, Prescott and Vitoria, *The Modern Law of Copyright and Designs*, 2nd ed (1995), vol 1 at 799-802.

34 In Campbell (ed), *Data Processing and the Law*, (1984) 209 at 212.

```
20 GOTO 20
```

38 If we view this set of instructions at a lower level of abstraction, in a lower level language known as Z-80 assembly language, however, the set of instructions would be:

```
ORG 7D00H
LD A,42
LD (3E20H),A
LOOP JP LOOP
END 7D00H
```

39 The following programming comments[35] provide a description of the Z-80 commands in terms of the corresponding actions of the microprocessor:

ORG 7D00H	Assemble program to start at address 7D00H
LD A,42	Load 42 (=ASCII for "*") into the A-register
LD (3E20H),A	Copy contents to address 3E20H (=screen centre)
LOOP JP LOOP	Keep jumping to the address "LOOP" ad infinitum, i.e. wait
END 7D00H	Entry point

40 If we express the same "set of instructions" in object code, which is at a still lower level of abstraction and expresses the contents of various "addresses" or "memory locations" in hexadecimal notation, the language would be:

Address	Contents
7D00	3E2A
7D02	32203E
7D05	C3057D

41 The object code, being at a low level of abstraction, is quite a close representation of the physical reality of what is occurring in the computer. Each address location is a label for a particular group of circuits in the computer. The hexadecimal number which sets out the content of the address expresses a binary number in hexadecimal notation. Each hexadecimal character 0-9 and A-F is the representation of a four-bit binary number in the range 0000 to 1111. The 0's and 1's themselves are a representation of the absence or presence, respectively, of

---

35 Laddie, Prescott and Vitoria, *The Modern Law of Copyright and Designs*, 2nd ed (1995), vol 1 at 836.

electrical impulses in that circuit. The logic functions which are hard wired into the computer operate on these electrical impulses to eventually produce the result of printing an asterisk on the screen.

42 In practice, when the above set of instructions in BASIC is typed into a computer, a separate program known as a compiler program reads the instruction "PRINT". It recognises that this is a command to invoke a program in object code which provides the computer with a set of electrical impulses to feed into its hard wired logic functions which will have the effect of printing a character on the screen. The choice of the word "PRINT" in BASIC is arbitrary in the sense that it would be possible to change the BASIC compiler program so that the typing of the word "TYPE" caused the same program in object code to be invoked as had been previously invoked for the word "PRINT".

43 No doubt, at the highest level of abstraction, the word "PRINT" is an expression of *an instruction* which is intended to cause a device having digital information processing capabilities to perform a particular function. Thus, at the highest level of abstraction, each of the Reserved Words in Dataflex may likewise be regarded as an expression of *an instruction* which is intended to cause a device having digital information processing capabilities to perform a particular function.

44 However, the appellant must show that each Reserved Word is an "expression, in any language, code or notation, *of a set of instructions* ... intended, either directly or after ... conversion to another language, code or notation ... to cause a device having digital information processing capabilities to perform a particular function"[36].

45 In order to overcome the difficulty that a Reserved Word is only one instruction in the Dataflex language, the appellant made two related arguments:

– first, that after "conversion to another language, code or notation" (i.e. source code), each Reserved Word is a "set of instructions" in source code and each Reserved Word is therefore the expression, in the Dataflex language, of this underlying set of instructions; and

– second, that each Reserved Word is an expression, in the Dataflex language, of, as the appellant put it, "the meaning and syntax of the word or command in question".

46 In relation to the first argument, it is true that after a Reserved Word is converted to source code, there is a "set of instructions" in source code.

---

36 Emphasis added.

47 The question then arises as to the level of abstraction (or in other words, the level of language) at which the appellant must show that the Reserved Word meets the definition of a computer program.

Two competing interpretations of "computer program"

48 In our view, there are two competing interpretations of the definition of "computer program" in s 10(1) of the Act. The two competing interpretations arise from the effect of the words "in any language, code or notation" and the words "either directly or after ... conversion to another language, code or notation" in the definition.

49 On one view, the effect of these words is that, if an item written in language A is not an "expression ... of a set of instructions ... intended ... to cause a device having digital information processing capabilities to perform a particular function" in language A, but after conversion to language B is such an expression of a set of instructions in language B, the statutory requirements are met in respect of the expression in both language A and language B. Thus, the item would be regarded as a "computer program" in both language A and language B, even though it is only an "expression ... of a set of instructions ... intended ... to cause a device having digital information processing capabilities to perform a particular function" in language B.

50 The second, opposing view is that the requirement that there be an "expression ... of a set of instructions ... intended ... to cause a device having digital information processing capabilities to perform a particular function" is a requirement that must be applied to each new language in which the item may be expressed in order for the item to be a "computer program" in that language. On this view, if an item is not an "expression ... of a set of instructions ... intended ... to cause a device having digital information processing capabilities to perform a particular function" in language A, its expression in language A is not a computer program, even if, after conversion to language B, the item meets the criteria of an "expression ... of a set of instructions ... intended ... to cause a device having digital information processing capabilities to perform a particular function" in language B.

51 On the second view, the question whether an item is an "expression ... of a set of instructions ... intended ... to cause a device having digital information processing capabilities to perform a particular function" must be determined separately for each language in which that item is expressed. The fact that an item meets the criteria of an "expression ... of a set of instructions ... intended ... to cause a device having digital information processing capabilities to perform a particular function" in one language does not automatically mean that the item will meet that criteria in any other language in which that item may be expressed.

52 In our opinion, the second view is the preferable one. The definition of "computer program" begins with the words "an expression, in any language, code

or notation". The phrase relates to a singular expression (the word "an" is used) and the words "any language" envisage that the expression will be in a particular language, whatever that language might be. However, the "expression" must be of a "set of instructions" which has a particular intention.

53 The meaning of the phrase "expression ... of a set of instructions" was referred to in the Explanatory Memorandum to the Copyright Amendment Bill 1984[37]:

> "The phrase 'expression ... of a set of instructions' is intended to make clear that it is not an abstract idea, algorithm or mathematical principle which is protected but rather a particular expression of that abstraction. The word 'set' indicates that the instructions are related to one another rather than being a mere collection."

54 It is the particular selection, ordering, combination and arrangement of instructions within a computer program which provide its expression. A computer program in a particular language may be relatively inefficient because it uses many instructions to achieve the function that a single instruction could achieve. A computer program in a particular language may also operate relatively inefficiently because of the way it is structured, in terms of the ordering of the instructions and the sequence in which they are executed. Considerations of efficiency are largely a function of the particular language which is used. It is the skill of the programmer in a particular language which determines the expression of the program in that language.

55 The Explanatory Memorandum states that it is a "*particular* expression" of an abstract idea which is protected[38]. As a particular expression is a function of the language of the expression, whether a word or words is or are a relevant expression of a set of instructions needs to be asked separately for each language in which there is purportedly a set of instructions.

56 For an item to be a computer program, it must not only be an "expression ... of a set of instructions", but the expression of that set of instructions must also be designed to achieve a particular purpose. That is to say, it must be "intended ... to cause a device having digital information processing capabilities to perform a particular function". The emphasis on a singular function in the phrase "a *particular* function" indicates that it is necessary to identify precisely the relevant function.

---

37 At par 16.

38 Explanatory Memorandum, Copyright Amendment Bill 1984, par 16 (emphasis added).

57 As we have already indicated, the only operations which a computer is physically capable of performing are those logic operations which are hard wired into its circuits. This physical limitation manifests itself at every level of computer programming, at different levels of abstraction. At the lowest level, when programming in object code, the limitation is perhaps most evident. That is because the computer program is written in terms which are closely related to physical events within the processor and memory of the computer. In higher level languages, the physical limitations of the computer manifest themselves to a programmer in a more subtle way. In the particular language in which a programmer is working, there is a limited set of commands which can be used. Each of these commands has its own syntactical and grammatical rules which must be followed in order for the command to be successfully recognised by the compiler program which converts the commands into object code. This is because each command in the high level language is nothing more than a "pre-packaged set" of sequences of the logic operations which the computer is capable of performing. The compiler program, upon reading a command, merely opens the pre-packaged set and launches the corresponding logic operations which the computer is capable of performing. If a set of instructions in a high level language is intended to cause a computer to perform a particular function, it is an expression which intends to express an algorithmic or logical relationship between the desired function and the physical capabilities of the computer, albeit indirectly. Owing to programming errors, or what are commonly called "bugs", it may not actually do so. The presence of "bugs" in a computer program, however, does not disentitle it to copyright protection, because as the Explanatory Memorandum stated[39]:

> "The phrase 'intended ... to cause' is used in preference to words such as 'capable ... of causing' to cover the situation where the program, as written, may not operate for technical reasons such as the presence of a programming error."

58 It is the ability to express in a computer language an algorithmic or logical relationship between an identifiable function which is desired to be performed and the physical capabilities of the computer, which is the true skill of the programmer. This remains true even if the programmer is working via the medium of a high level language and is unaware of the physical capabilities of the computer. It is the expression of this skill which is intended to be protected by the Act.

59 In our opinion, the foregoing conclusion also explains the reference in the definition of "computer program" to "a set of instructions (whether with or without any related information)". The Explanatory Memorandum stated[40]:

---

39 Explanatory Memorandum, Copyright Amendment Bill 1984, par 19.

40 Explanatory Memorandum, Copyright Amendment Bill 1984, par 18.

"The phrase 'whether with or without related information' is intended to make clear that the protected program may include material other than instructions for the computer (such as information for programmers or users of the program, or data to be used in connection with the execution of the program)."

60 The distinction between "data" or "related information" on the one hand and "instructions for the computer" on the other indicates that Parliament has conceived of a "set of instructions" that are truly "instructions for the computer" in the sense that they are referable to the computer's physical capabilities. The inclusion in the definition of "computer program" of the words "(whether with or without related information)", in parenthesis, effected the legislative intent that the inclusion of "related information" within the "expression ... of a set of instructions" would not take that expression outside the definition. "Data" or "related information" is that part of the computer program which is not in any sense referable to the computer's physical capabilities. The examples given in the Explanatory Memorandum indicate that it may be a wide category. It is unnecessary to consider what may constitute "information" which is not "related information", a matter removed from the task of construing the phrase "set of instructions".

61 In our opinion, whether what is claimed to be a "computer program" is an "expression ... of a set of instructions ... intended ... to cause a device having digital information processing capabilities to perform a particular function" must be answered separately for each language in which the item in question is said to be a computer program.

62 Moreover, something is not a "computer program" within the meaning of the definition in s 10(1) unless it intends to express, either directly or indirectly, an algorithmic or logical relationship between the function desired to be performed and the physical capabilities of the "device having digital information processing capabilities". Thus, in the sense employed by the definition, a program in object code causes a device to perform a particular function "directly" when executed. A program in source code does so "after ... conversion to another language, code or notation".

63 Some support, by way of analogy, may be derived from considering the position in the United States. In *Baystate Technologies Inc v Bentley Systems Inc*[41], it was held that whilst the computer program comprising "CADKEY" was protected, the particular "data structures"[42] with which the case was concerned

---

41 946 F Supp 1079 (1996).

42 These are "constructs that allow a programmer to combine a variety of different types of data into manageable units. For example, an employee record consisting of the employee's name, address, social security number and salary rate could be

(Footnote continues on next page)

"[did] not bring about any result on their own", so that they were protected, if at all, only as part of the whole computer program[43]. This was because, as was later expressed in the judgment[44]:

> "a computer cannot read data structures and perform any function".

64 Once these principles are applied to each Reserved Word in the Dataflex language, it is clear that they are not "computer programs". Each Reserved Word comprises but a single instruction in that language. Each Reserved Word, considered alone, is not a "set of instructions" in that language. It is not a "computer program" expressed in the Dataflex language.

Meaning and syntax

65 There remains to be addressed the further argument of the appellant that the relevant set of instructions at the level of the Dataflex language is the "meaning and syntax of the word or command in question". In response to questions from members of this Court during the argument of the appeal, counsel for the appellant was asked on a number of occasions to identify the relevant "set of instructions". His answer was that it was the "meaning and syntax of the word or command in question".

66 However, the function which will be executed by a particular Reserved Word depends entirely on the source code underlying the Reserved Word. Thus, its "meaning" depends on the source code underlying it. This is also the case with the "syntax". There are, of course, grammatical and syntactical rules for the use of the Reserved Words. These rules would be written in the source code underlying the Dataflex commands. Equivalent rules would be written in the source code underlying the PFXplus commands. The meaning of the Reserved Words, and the grammatical and syntactical rules for their use, are not an "expression" until they are reduced to the underlying source code. However, the appellant does not, and could not, contend that Dr Bennett's source code expression of the meaning of commands or of the grammatical and syntactical rules in PFXplus is a reproduction of the source code expression of those meanings or rules in the Dataflex language. There was a finding that the source code of PFXplus was dissimilar to the source code of Dataflex.

67 Furthermore, as the Full Court of the Federal Court pointed out[45]:

---

combined into a data structure that the program could manipulate as a single unit": *Nimmer on Copyright*, vol 4, §13.03[F][1], n 288.

43 946 F Supp 1079 at 1086 (1996).

44 946 F Supp 1079 at 1090 (1996).

"It goes without saying that, but for ease of usage, the precise words used in a particular computer language are irrelevant. A particular program could use the letters 'XZB', or any other combination of letters or symbols, so that when the computer was confronted with those letters it would perform a particular set of instructions, for example, displaying a particular item in a particular database. The use of a string of letters forming an English word such as 'display' will, however, more easily be recalled and reproduced by the user than will a meaningless combination to that user."

68 Thus, from the computer's perspective, any of the Reserved Words in the Dataflex language could have been replaced with any other word or string of characters. If it had, the same function would have been performed provided the necessary modifications were made to the compiler and runtime programs. In our opinion, this shows that the particular characters of a Reserved Word in the Dataflex language, considered alone, do not intend to express a logical or algorithmic relationship between the function it intends to cause the computer to perform and the physical capabilities of the computer.

69 It is true that, from the user's perspective, the particular words chosen allow for ease of use. No doubt this explains the focus in the appellant's submissions on the "meaning" of a Reserved Word. However, ease of use is an intended function which the author of the Reserved Words had in relation to the user. It is not an intended function which the author of the Reserved Words had in relation to the computer. The definition of a computer program requires that the intention be in relation to the performance of a particular function by the computer.

70 It is true that in any high level language, the particular commands might be replaced with any other string of characters and necessary modifications made to the compiler program. But that does not mean that nothing expressed in a high level language could ever be a computer program. Once more than one instruction is expressed in a high level language with the intention that the expression will, after conversion to object code, cause a computer to perform a particular function, there will ordinarily be a computer program for the purposes of the Act. The choice and interrelationship of the particular instructions used and their sequence and structure will ordinarily constitute the expression of a logical or algorithmic relationship between the function intended to be performed and the physical capabilities of the computer.

71 The conclusion that the Reserved Words themselves are not a computer program in Dataflex does not mean that their expression in source code and object code is not a computer program. As the Full Court stated[46], correctly in our view:

---

45 (1997) 75 FCR 108 at 114.

46 (1997) 75 FCR 108 at 122.

"[I]t is in our view not accurate to refer to each of the words as being an expression of the set of instructions. The set of instructions is expressed in the source code which is the computer program and, at least at a higher level, includes the particular word which is a command. The computer program will also in other forms exist in lower level language, ultimately through to an object code in non-visible form. Each of these representations will fall within the definition of a 'computer program'. In each of them, in some language, code or notation, the word said to be part of the computer language will be able to be found."

72 However, the appellant does not contend that the source code in PFXplus underlying any of the 166 commands in question is an infringement of the source code underlying the corresponding commands in Dataflex. That being so, the claim that each Reserved Word is a computer program fails.

2. *Is the collocation of the Reserved Words a computer program?*

73 Furthermore, the collocation of the Reserved Words is not a "computer program". Although the Reserved Words together form "an expression ... of a set of instructions", their simple listing together, without more, does not cause a computer to perform any identifiable function. There is no interrelationship of the instructions with one another which is an expression of a logical or algorithmic relationship between an identifiable function and the physical capabilities of the computer via the medium of the Dataflex language.

74 It is no answer that there is a set of instructions with a single identifiable function in that it provides a programmer with the vocabulary to enable him or her to program in the Dataflex language. As in the case of each individual Reserved Word, this is a function which the author of the Reserved Words intended them to perform in relation to the user, not in relation to the computer. As we have indicated, the definition of a "computer program" requires that the set of instructions be intended to cause the computer to perform a particular function.

3. *Does the collocation of the Reserved Words form a substantial part of a literary work (the Dataflex system)?*

75 The Dataflex system is a computer program. Hence it is a literary work for the purpose of the Act. The appellant contends that the collocation of Reserved Words, even if it is not itself a literary work, constitutes a substantial part of the Dataflex system. Section 14(1)(b) of the Act provides that in the Act "a reference to a reproduction ... of a work shall be read as including a reference to a reproduction ... of a substantial part of the work". A copyright owner's

exclusive right to reproduce a work is therefore infringed[47] if another person reproduces a "substantial part" of the work.

76 In *Autodesk No 1*[48], this Court held that it was not necessary that the reproduction of a substantial part of a computer program should itself be a computer program. Relying on that reasoning, the appellant contends that its copyright is infringed because the collocation of Reserved Words is a substantial part of the Dataflex system.

Substantiality

77 The question whether something is a substantial part of a computer program created difficulty in *Autodesk No 1*[49] and *Autodesk No 2*[50]. In *Autodesk No 1*, in the course of determining whether the 127-bit series embedded in the EPROM in Dyason's Auto Key lock infringed the copyright in Autodesk's Widget C, Dawson J, with whom the other members of the Court agreed, said[51]:

> "For Widget C is a computer program and a substantial, *indeed essential,* part of that program is the look-up table by reference to which Widget C processes the information which it receives from the AutoCAD lock. ... In effect, both Widget C and the Auto Key lock contain the same look-up table.
>
> ... Whilst the 127-bit look-up table does not of itself constitute a computer program within the meaning of the definition – it does not by itself amount to a set of instructions – it is a substantial part of Widget C and its reproduction in the Auto Key lock is a reproduction of a substantial part of that program." (emphasis added)

78 In *Autodesk No 2*[52], when discussing whether the 127-bit series embedded in the EPROM constituted a reproduction of a substantial part of Widget C, Brennan J said:

---

47 Section 31(1)(a)(i) and s 36(1) of the Act.

48 (1992) 173 CLR 330.

49 (1992) 173 CLR 330.

50 (1993) 176 CLR 300.

51 (1992) 173 CLR 330 at 346.

52 (1993) 176 CLR 300 at 311-312.

"A further submission is that the look-up table is not a substantial part of the relevant computer program. The bytes contained in the look-up table are but a minute fraction of the bytes in the whole of the Widget C program. Nevertheless, the series of digits in the look-up table is both original and critical to the set of instructions designed to cause the computer to run the AutoCAD application. When the running of the AutoCAD application is the purpose of the set of instructions expressed in that program, it would be difficult, if not impossible, to contend that the look-up table is not a substantial part of that program."

79 Gaudron J said[53]:

"The only other matter on which the respondents rely is the question whether the look-up table can be said to be a substantial part of Widget C. In truth, the table was the linchpin of the program, to borrow an expression from a different technology. It was the critical part of the instructions in that the other parts depended on and were made by reference to it. Whatever may be the situation in cases in which information plays a less significant role, given that the look-up table was crucial to Widget C and given that copyright protection extends to information as well as the commands involved in a set of instructions of the kind constituting a computer program as defined in s 10 of the Act, there is, in my view, simply no basis for an argument that the look-up table was not a substantial part of Widget C."

80 The above passages in the various judgments in the two *Autodesk* cases relied on the "essentiality" or "criticality" of the look-up table to determine its substantiality. The reasoning appears to come close to a "but for" analysis, i.e. but for the look-up table, the AutoCAD program would not execute and therefore the look-up table was a "substantial part" of the program. To some degree the course taken in the above passages may reflect the statement by Dawson J[54] that, although the 127-bit look-up table did not itself constitute a computer program within the meaning of the definition because it did not by itself amount to a set of instructions, so that no reliance was placed upon it as a literary work in itself, "there can be no doubt about the originality of authorship of the look-up table expressed as it is in Widget C"[55]. This may have suggested that what was "original" ought itself to be protected.

---

53 (1993) 176 CLR 300 at 330.

54 *Autodesk No 1* (1992) 173 CLR 330 at 346-347.

55 *Autodesk No 1* (1992) 173 CLR 330 at 347.

81 However, as Mr Prescott QC has said[56]:

> "In general, a computer program – any computer program – will not work if even one digit therein is altered or corrupted. It is therefore 'essential'. But it would be a startling conclusion to hold that not even a single number from a computer program may be copied. If one digit will not infringe, why should 127? Only if it takes substantially more skill and labour to create the 127. But it does not."

82 In *Cantor Fitzgerald International v Tradition (UK) Ltd*[57], Pumfrey J, sitting in the English Patents Court, observed of the reasoning in *Autodesk No 1* that it "would result in any part of any computer program being substantial since without any part the program would not work, or at best not work as desired"[58].

83 In *Autodesk No 2*, Mason CJ, who dissented, took a different view of the substantiality issue. His Honour thought that the judgment in *Autodesk No 1* should be re-opened in order to hear the respondent's argument as to whether the look-up table was a substantial part of Widget C. Mason CJ said[59]:

> "It is clear that the phrase 'substantial part' refers to the quality of what is taken rather than the quantity[60]. In *Ladbroke (Football) Ltd v William Hill (Football) Ltd*, Lord Pearce stated[61]:
>
> > 'Whether a part is substantial must be decided by its quality rather than its quantity. The reproduction of a part which by itself has no originality will not normally be a substantial part of the copyright and therefore will not be protected. For that which would not attract copyright except by reason of its collocation will, when robbed of that

---

56 Prescott, "Was AutoCAD Wrongly Decided?", (1992) 14(6) *European Intellectual Property Review* 191 at 194. See also Kremer, "Before the High Court", (1998) 20 *Sydney Law Review* 296 at 306-307.

57 Unreported, High Court of Justice, Chancery Division, 15 April 1999.

58 Unreported, High Court of Justice, Chancery Division, 15 April 1999 at [75].

59 (1993) 176 CLR 300 at 305.

60 *Hawkes & Son (London) Ltd v Paramount Film Service Ltd* [1934] Ch 593; *Ladbroke (Football) Ltd v William Hill (Football) Ltd* [1964] 1 WLR 273; [1964] 1 All ER 465; *Greenfield Products Pty Ltd v Rover-Scott Bonnar Ltd* (1990) 95 ALR 275 at 293; 17 IPR 417 at 436; *Dixon Investments Pty Ltd v Hall* (1990) 18 IPR 481.

61 [1964] 1 WLR 273 at 293; [1964] 1 All ER 465 at 481.

> collocation, not be a substantial part of the copyright and therefore the courts will not hold its reproduction to be an infringement. It is this, I think, which is meant by one or two judicial observations that "there is no copyright" in some unoriginal part of a whole that is copyright.'
>
> As this statement makes clear, in determining whether the quality of what is taken makes it a 'substantial part' of the copyright work, it is important to inquire into the importance which the taken portion bears in relation to the work as a whole: is it an 'essential' or 'material' part of the work[62]?
>
> In this case, it is argued by the appellants that such an inquiry compels an affirmative answer as the look-up table is essential to the operation of the AutoCAD locking mechanism. Such an argument, however, misconceives the true nature of the inquiry and seeks to re-introduce by another avenue an emphasis upon the copyright work's function. True it is that the look-up table is essential to the functioning of the AutoCAD lock. However, in the context of copyright law, where emphasis is to be placed upon the 'originality' of the work's expression, the essential or material features of a work should be ascertained by considering the originality of the part allegedly taken. This is particularly important in the case of functional works, such as a computer program, or any works which do not attract protection as ends in themselves (e.g., novels, films, dramatic works) but as means to an end (e.g., compilations, tables, logos and devices)[63]."

84 There is great force in the criticism that the "but for" essentiality test which is effectively invoked by the majority in *Autodesk No 2* is not practicable as a test for determining whether something which appears in a computer program is a substantial part of it. For that reason, we prefer Mason CJ's opinion that, in determining whether something is a reproduction of a substantial part of a computer program, the "essential or material features of [the computer program] should be ascertained by considering the originality of the part allegedly taken"[64].

85 In order for an item in a particular language to be a computer program, it must intend to express, either directly or indirectly, an algorithmic or logical relationship between the function desired to be performed and the physical capabilities of the "device having digital information processing capabilities". It follows that the originality of what was allegedly taken from a computer program must be assessed with respect to the originality with which it expresses that

---

62 Ricketson, *The Law of Intellectual Property*, (1984) at 169.

63 See, e.g., *Mirror Newspapers Ltd v Queensland Newspapers Pty Ltd* [1982] Qd R 305; *Klissers Farmhouse Bakeries Ltd v Harvest Bakeries Ltd (No 2)* [1985] 2 NZLR 143 at 157; (1985) 5 IPR 533 at 548.

64 (1993) 176 CLR 300 at 305.

algorithmic or logical relationship or part thereof. The structure of what was allegedly taken, its choice of commands, and its combination and sequencing of commands, when compared, at the same level of abstraction, with the original, would all be relevant to this inquiry.

86 That being so, a person who does no more than reproduce those parts of a program which are "data" or "related information" and which are irrelevant to its structure, choice of commands and combination and sequencing of commands will be unlikely to have reproduced a substantial part of the computer program. We say "unlikely" and not "impossible" because it is conceivable that the data, considered alone, could be sufficiently original to be a substantial part of the computer program.

87 It follows that we are unable to agree with the approach to determining "substantiality" which the majority took in *Autodesk No 1* and *Autodesk No 2*. Because of the importance of the question, we think that the Court should re-open the question of what constitutes a substantial part of a computer program. To depart from the reasoning in the *Autodesk* cases does not necessarily mean that the outcomes in those cases were wrong. In our view, the look-up table in Widget C was merely data and was not capable of being a substantial part of the AutoCAD program unless the data itself had its own inherent originality. However, re-opening the reasoning in the *Autodesk* cases does not require the Court to express a view on whether the look-up table in that case had its own inherent originality.

Substantiality and the collocation of the Reserved Words

88 As they appear in the source code of the Dataflex system, the Reserved Words are irrelevant to the structure, choice of commands and combination and sequencing of the commands in source code. They are merely literal strings which, from the computer's perspective, could be replaced by any other literal string. Accordingly, they are not a substantial part of the Dataflex program as it appears in source code unless they have their own inherent originality.

89 The evidence is that at least 55 of the Reserved Words are unique to the Dataflex program. When looking at the list of the Reserved Words in respect of which the respondent was held at first instance to be infringing copyright, it can be seen that many of the words are ordinary English words which are suggestive of the function they perform, such as "BOX"; "CHART"; "CHECK"; "CLEAR"; "INDICATOR"; "INSERT"; "LOOP"; "NAME"; "PAD"; "PALETTE"; "PLOT"; "POINTS"; "REQUIRED"; "RETAIN"; "SELECTION"; "STATUS"; "TOTAL" and "UNLOCK".

90 Others are concatenations of two or more English words which together suggest the function performed, such as "AUTOFIND"; AUTOPAGE"; AUTORETURN"; "BACKFIELD"; "PAGEBREAK"; "PAGECHECK";

"PAGECOUNT"; "SAVE_GRAPHIC"; "SET_LINE_STYLE"; "SKIPFOUND" and "WINDOWINDEX".

91 Still other Reserved Words are single words which are in common use in other computer languages such as "DIRECTORY" and "SAVE", or concatenations of other words which are common computer terms such as "FILELIST", "MAKE_FILE", "RUNPROGRAM", and "SAVERECORD". As the Full Court pointed out[65], even those Reserved Words which are more removed from English usage than the examples above, such as "ENTAGAIN", "KEYPROC", and "MOVEINT", have a function which can be guessed from the English association.

92 In our opinion, even when the Reserved Words are considered as a collocation, they do not possess sufficient originality as data to constitute a substantial part of the computer program which is the Dataflex system.

4. *Does copyright subsist in the table or compilation of the Reserved Words in the Dataflex User's Guide?*

93 The Reserved Words are contained in the Dataflex User's Guide. The appellant did not submit that any of the Reserved Words themselves were traditional literary works protected by copyright, no doubt because they would face significant hurdles in the form of originality and substantiality. Given that the Reserved Words are arranged in alphabetical order in the Dataflex User's Guide, very little skill or labour was involved in compiling the Reserved Words in the form in which they appear in the User's Guide over and above the sum of the skill and labour involved in devising each individual Reserved Word. As the Full Court said[66]:

> "This is not a case where disconnected words are used in a particular order so that the order becomes the linchpin for copyright."

94 Furthermore, as we have already said, each of the Reserved Words is suggestive of the function it performs. In many cases, it is an ordinary English word, or a concatenation of two or more ordinary English words.

95 Even if the skill and labour involved in devising each individual Reserved Word is combined and consideration given to the total skill and labour, there may still be a real question as to whether there is sufficient originality for copyright to

---

65 (1997) 75 FCR 108 at 114.

66 (1997) 75 FCR 108 at 124.

subsist in the combination[67]. This is so even allowing for the inclusion in the definition of par (b) of "literary work" of a "compilation of computer programs".

96 The totality of the Reserved Words cannot be protected as a "compilation" within the definition because it requires a "compilation of computer programs" and the Reserved Words are not themselves programs. This does not necessarily preclude them together from protection as constituting a single program, but the set of instructions said to constitute such a program would still require identification. For the reasons leading to the conclusion that each of the Reserved Words does not constitute programs, a collection thereof does not constitute a program. The English letters which make them up are never at any stage executed by the computer. They are not instructions. They never cause a computer to perform a function. Their totality might be considered a "set", but of labels or data, rather than of instructions as required by the definition.

97 In any event, even if copyright does subsist in the table or compilation of the Reserved Words, we do not think that the respondents have infringed this copyright. The Reserved Words appear in the PFXplus source code program not as an alphabetical list, but as literal strings to which certain commands are assigned.

B. THE MACROS

98 The appellant also contends that copyright subsists in what are referred to as "Macros". It contends that each of these commands is a "computer program" within the statutory definition and that Dr Bennett made an adaptation of the Dataflex Macro commands. Consequently, it submits that Dr Bennett infringed the appellant's copyright in them.

99 Three particular commands in the Dataflex language, "REPORT", "ENTERGROUP" and "ENTER", are described as "Macros" because they cause the performance of a more complex function than any of the other Reserved Words. Executing a Macro command causes a sequence of other functions to be executed, so that the overall effect of performing a more complex function is achieved.

Are the Macros computer programs in Dataflex?

100 It follows from the nature of a "computer program" as defined in s 10(1) of the Act that the words assigned to the Macros, comprising as they do one instruction in the Dataflex language, cannot qualify as a "computer program".

---

67 cf *G A Cramp & Sons Ltd v Frank Smythson Ltd* [1944] AC 329; *Australian Consolidated Press Ltd v Morgan* (1965) 112 CLR 483 at 486-487; *Exxon Corporation v Exxon Insurance Consultants International Ltd* [1982] Ch 119 at 144.

However, the underlying source code of each Macro may qualify as a "computer program". In practice, the source code underlying each Macro is a small fragment of the source code of the overall Dataflex computer program (the relevant portion was said by the Full Court of the Federal Court to be some 229 lines).

101 The Full Court said[68] that the question of whether a component part of a computer program is itself a computer program for the purposes of the Act is a question of fact. However, the Full Court went on to say that "[i]f a particular set of instructions is functionally separate from the entirety of the program, then ... there is no difficulty in treating that set of instructions as being a literary work separate from the balance of the program". Although it did not expressly say so, the Full Court must have considered that the particular set of instructions comprising each Macro was not functionally separate from the remainder of the Dataflex compiler program. This is because it said[69] that "the relevant program to be considered here would not be that small fragment of program which causes the macro command to perform its function (some 229 lines), but the Dataflex compiler program itself".

102 The Full Court stated no statutory basis for taking a "functionally separate" approach. If there is a statutory basis it must be found in the words of the definition "intended ... to cause a device having digital information processing capabilities to perform a particular function". If the segment of the larger program in question can be said to have a function which is not merely one of the functions in the set of functions performed by the larger program, but is a separate and distinct *particular* function, it may be that this segment can be properly viewed as a computer program in and of itself. The *Autodesk* example given by the Full Court[70] would illustrate that kind of distinction. The lock program in Widget C had a function (to prevent use of the software unless the correct key was inserted) which was not merely one of the functions in the set of functions performed by the computer aided design program which was AutoCAD.

103 The Full Court appears to have concluded in this case that as a matter of fact the function performed by the segment of the Dataflex source code which underlies each Macro is merely one of the functions performed by the larger program, and therefore that segment is not a computer program in and of itself. We need not consider that conclusion because even if it is not right and the segment of source code which underlies each Macro is a computer program, there was no reproduction and no adaptation of those works.

---

68 (1997) 75 FCR 108 at 126-127.

69 (1997) 75 FCR 108 at 127.

70 (1997) 75 FCR 108 at 127.

Reproduction or adaptation of the Macros?

104 The learned trial judge found strong objective similarity between the underlying source code of the PFXplus Macros and the underlying source code of the Dataflex Macros. Jenkinson J did not find that Dr Bennett had reproduced the Dataflex Macros, but instead found that Dr Bennett had "made an adaptation of the expression in I-Code of each of the three sets of instructions"[71]. The finding of no reproduction was not challenged; the appellant contended that there was an adaptation.

105 The meaning of "adaptation" in relation to computer programs, as set out in s 10(1) of the Act, is "a version of the work (whether or not in the language, code or notation in which the work was originally expressed) not being a reproduction of the work". In examining the meaning of the word "version", the Full Court referred[72] to the meanings of the word "version" given by the *Macquarie Dictionary*[73] "2. a translation. 3. a particular form or variant of anything". The Full Court also quoted[74] the following passages from the Explanatory Memorandum[75]:

> "11. Copyright in literary works includes exclusive rights to reproduce or adapt such works and computer programs will be treated as literary works. However, the present definition of adaptation in relation to literary works only includes translation, conversion between dramatic and non-dramatic forms, and conversion to a pictorial form.
>
> 12. Of these, only translation is likely to be relevant to adaptation of programs and there are legal doubts as to whether this refers only to translations between human languages.
>
> 13. The new definition is intended to cover translation either way between the various so-called 'high-level programming languages' in which the programs may be written by humans (often called 'source code') and languages, codes or notations which actually control computer operations (often called 'machine code' or 'object code'). Thus 'adaptation' is intended, for example, to cover the compilation of a

---

71 (1996) 63 FCR 336 at 344. "I-Code" is an intermediate level of code between the Dataflex level and the source code level.

72 (1997) 75 FCR 108 at 125.

73 2nd ed (1991) at 1938.

74 (1997) 75 FCR 108 at 125-126.

75 Explanatory Memorandum, Copyright Amendment Bill 1984.

FORTRAN program to produce machine code which will directly control the operation of a computer. Languages, etc of intermediate level would also be covered.

14. It is also possible for a program to be converted from object code into source code, or between different languages of similar level. In some circumstances this process will result largely in a substantial reproduction of the original program. In other cases, however, such as compilation followed by de-compilation, the differences may be so substantial that one cannot speak of a reproduction although the final product is clearly derived from the original. The new definition of adaptation is intended to cover such situations."

106 The Full Court said[76]:

"The evidence is clear that while Dr D Bennett carefully studied the Dataflex program so as to ensure that the PFXplus commands in question performed the same functions as the Dataflex commands, the expression of the source program as written by him was an original expression, albeit having much which was objectively similar to the expression of the source code in the Dataflex program. But it is clear that the process involved no translation from one form or language to another, nor did it involve the kind of process referred to in par 14 of the Explanatory Memorandum involving compilation followed by decompilation, or vice versa. In our view, a process of devising a source code to perform the same function as is performed in some other source code expressed in original language does not involve creating a version of the original source code."

107 Thus, the Full Court was of the opinion that there needed to be "translation from one form or language to another", or, alternatively, "the kind of process referred to in par 14 of the Explanatory Memorandum involving compilation followed by decompilation, or vice versa" in order for there to be a "version" within the meaning of the statute.

108 In response to this reasoning, the appellant contends that while the word "version" includes these two processes, it is not limited to these two processes. Furthermore, the appellant argues that the statutory context does not support the approach of the Full Court. It points out that where, in the course of the amendments to the Act that included the definition of "adaptation", Parliament wished to refer to conversion of a program from one language, code or notation to another, it used the word "conversion". The definition of "computer program" contains one example. Furthermore, so the appellant submits, the word "version" is ordinarily used in a much looser sense than that adopted by the Full Court. Ordinarily, it refers to the substance of the work but not the form. Examples are

---

76 (1997) 75 FCR 108 at 126.

rendering a novel into a play or vice versa or turning a story into a series of pictures.

109 Paragraph 12 of the Explanatory Memorandum states that "only translation is likely to be relevant to adaptation of programs". This indicates that Parliament did not intend the word "version" to cover situations where, although the functionality of a computer program was copied, original code has been written to perform that function. The focus on translation indicates that Parliament was concerned to ensure that the different languages in which a computer program may be expressed did not provide a means by which copying could occur and infringement be avoided on the ground that the expression in the new language was not a "reproduction".

110 The use of the words "derived from the original" in par 14 of the Explanatory Memorandum also indicates that the focus is on copying. In accordance with the fundamental principle that copyright protects expression and not ideas, this must relate to the copying of the code (the "expression ... of a set of instructions"), rather than a copying of the idea or function underlying the code.

111 There was no adaptation of the Macros.

C. THE DATAFLEX HUFFMAN COMPRESSION TABLE

112 The respondents seek special leave to cross-appeal against the Full Court's finding that they infringed the copyright which the appellant held in the Huffman compression table embedded in the Dataflex program.

113 Usually, in storing data, all of the 256 characters which a computer recognises are stored in memory as bit strings which are eight bits in length. Huffman compression is a method of reducing the amount of memory space consumed by data files. It stores characters in a data file as bit strings which have a length which relates to the character's frequency of occurrence in the data file. If a character occurs frequently in the data file, it is stored as a bit string of shorter length than a character which occurs infrequently in a data file.

114 The following example from Dr Bennett's evidence explains the method:

> "For example, the letter 'e' is normally encoded as the bit string '01100101', but as a common character it might be encoded instead as the bit string '101', a saving of 62.5%. The character '@', normally encoded as the bit string '01000000', occurs infrequently and might be encoded instead as the bit string '00100100101110'."

115 The Huffman algorithm, when expressed in source code, analyses a data file to determine the relative frequency of the occurrence of characters, and then assigns a bit string of appropriate length to each character, depending on its

frequency of occurrence. There is no allegation in this case that Dr Bennett copied the source code of the Huffman algorithm from the Dataflex program. Dr Bennett states that he obtained "freely distributable" source code for this purpose.

116 In about 1992, Mr Cory Casanave created the "standard" or "default" Dataflex Huffman compression table by writing a program which applied the Huffman algorithm to a database file known as SERIAL.DAT. There is also provision in the Dataflex program for users to create their own custom Huffman compression tables. This enables the user to compress the data in his or her files with reference to the frequency of occurrence of characters in the actual file to be compressed, rather than with reference to the frequency of occurrence of characters in the file SERIAL.DAT. It appears that SERIAL.DAT simply served the purpose of being a representative sample of data which would suffice for standard compressions. If the user wanted the greater efficiency which would flow from the compression table actually being derived from the data in the file to be compressed, custom compression could be used.

117 Dr Bennett wished PFXplus to be able to compress and decompress Dataflex files, and vice versa, when standard compression was used. Consequently, he needed to be able to replicate precisely the default Huffman compression table used in Dataflex. Dr Bennett did not have access to the file SERIAL.DAT. For that reason, he could not replicate the Huffman table simply by applying the Huffman algorithm to that file. His evidence is that he refrained from "decompiling or looking inside the Dataflex runtime", and instead carried out the following process:

> "The process I eventually devised was as follows. First I obtained a program which would dump out the contents of a file in bits, rather than the more usual hex dump.
>
> I then wrote a small Dataflex program which would create an empty database file using Standard compression, and would then add 256 records to the file. The records had the following appearance:
>
> AEAEAEAEAEAEAEA!AEAEAEAEAEAEAAEA
> AEAEAEAEAEAEAEA@AEAEAEAEAEAEAAEA
> AEAEAEAEAEAEAEA#AEAEAEAEAEAEAAEA
> AEAEAEAEAEAEAEA$AEAEAEAEAEAEAAEA
> AEAEAEAEAEAEAEA%AEAEAEAEAEAEAAEA
>
> and so on, for a total of 256 records.
>
> The position in the middle could be occupied by one of 256 possible characters, and the program cycled through all 256 possibilities. I knew that the pattern of AE repeated would create a distinctive and reproducible bit pattern in the Huffman encoding, to act as a background. It would then

be possible to make out the single changed character, standing out in relief against the background.

I ran the program, dumped the output to a file using the bit dump program, and printed out the result. I then marked out the 256 bit strings by pencil. ... I found that they did indeed stand out well against the background. This process took just a few hours."

118 By this process, Dr Bennett deduced the bit string for each of the 256 characters as it appeared in the standard Dataflex compression table. This enabled him to create an identical table for use in PFXplus.

119 The definition of "literary work" in the Act includes:

"a table, or compilation, expressed in words, figures or symbols (whether or not in a visible form)".

120 The Explanatory Memorandum to the Copyright Amendment Bill 1984 stated[77]:

"By removing the requirement that tables or compilations be in a visible form it is made clear that a computerised data bank, for example, may be treated as a compilation being a literary work. It is also important because data is often stored in a computer as a table. These changes are consistent with the definition of material form".

121 In our opinion, the Dataflex Huffman table is a table expressed in figures and symbols, and falls squarely within the statutory definition of a "literary work". The reference in the Explanatory Memorandum to "data ... stored in a computer as a table" clearly describes the Dataflex Huffman table. The Dataflex Huffman table is similar to the look-up table in Widget C which, in *Autodesk No 1*, Dawson J considered was a "literary work" within the meaning of the above definition. His Honour thought this was so even though no reliance was placed on that point by Autodesk[78].

122 For copyright to subsist in the standard Dataflex Huffman table, it must be an "original literary ... work"[79]. As we have indicated, the requirement that a work be "original" in copyright law is a requirement that "the work emanates from the person claiming to be its author, in the sense that he has originated it or

---

77 At par 26.

78 (1992) 173 CLR 330 at 347.

79 Section 32(1) of the Act.

brought it into existence and has not copied it from another"[80]. At first instance, Jenkinson J found that "[t]he use of the Huffman system to produce a compression table requires the employment of substantial skill and judgment and a very great deal of hard work"[81]. The Full Court agreed with this finding[82].

123 The skill and judgment employed by Dataflex was perhaps more directed to writing the program setting out the Huffman algorithm and applying this program to a representative sample of data than to composing the bit strings in the Huffman table. Nevertheless, the standard Dataflex Huffman table emanates from Dataflex as a result of substantial skill and judgment. That being so, the Full Court was correct in holding that the standard Dataflex Huffman table constituted an original literary work.

124 In addition, in our opinion the Full Court was correct in holding that the process undertaken by Dr Bennett constituted a "reproduction" of the standard Dataflex Huffman table. The fact that Dr Bennett used an ingenious method of determining the bit string assigned to each character does not make the output of such a process any less a "reproduction" than if Dr Bennett had sat down with a print-out of the table and copy-typed it into the PFXplus program.

125 The finding that the respondents infringed the appellant's copyright in the Huffman table embedded in the Dataflex program may well have considerable practical consequences. Not only may the finding affect the relations between the parties to these proceedings, it may also have wider ramifications for anyone who seeks to produce a computer program that is compatible with a program produced by others. These are, however, matters that can be resolved only by the legislature reconsidering and, if it thinks it necessary or desirable, rewriting the whole of the provisions that deal with copyright in computer programs.

Orders

126 The appeal should be dismissed with costs. Special leave to cross-appeal should be granted but the cross-appeal should be dismissed with costs. The costs should be set off.

GAUDRON J:

---

80 Ricketson, *The Law of Intellectual Property*, (1984) at 83.

81 *Data Access Corporation v Powerflex Services Pty Ltd* (1996) 63 FCR 336 at 344-345.

82 *Powerflex Services Pty Ltd v Data Access Corporation (No 2)* (1997) 75 FCR 108 at 128.

127 Subject to one matter, I agree with the joint judgment of Gleeson CJ, McHugh, Gummow and Hayne JJ. The matter upon which I hold a different view from their Honours relates to their analysis of the decisions in *Autodesk Inc v Dyason*[83] and *Autodesk Inc v Dyason [No 2]*[84]. Those cases were not simply concerned with the question whether, in words used in the joint judgment, "something which appears in a computer program is a substantial part of it"[85]. Although the material in question in the *Autodesk* cases, a look-up table, was not, when viewed in isolation, a set of instructions, it was, when viewed in context, part of the set of instructions constituting the computer program in question. It was not simply "data". Nor was it simply "information". Rather, as I pointed out in *Autodesk [No 2]*, it was an integral part of the set of instructions in that "other parts depended on and were made by reference to it"[86].

128 It is not necessary in this case to decide whether information which is not part of a set of instructions might, nonetheless, be a substantial part of a computer program. However, if that is to be, the information must be "related information", as that expression is used in the definition of "computer program" in s 10(1) of the Act. By that sub-section:

> "***computer program*** means an expression, in any language, code or notation, of a set of instructions (whether with or without related information) intended ... to cause a device having digital information processing capabilities to perform a particular function".

129 The definition of "computer program" posits a relationship between information and a set of instructions, although the precise nature of the relationship is not indicated. As already noted, there is a relationship of the requisite kind if the information, itself, forms part of the instructions. On the other hand and contrary to what is suggested in the joint judgment, information is not, in my view, "related information" for the purposes of the definition if it is "irrelevant to [the] structure [of the computer program], [its] choice of commands [or] combination [or] sequencing of commands"[87]. Information of that kind is not related in any relevant sense to any set of instructions. And if not related, it is impossible, in view of the definition, to treat that "data" or "information" as part of a computer program.

---

83 (1992) 173 CLR 330.

84 (1993) 176 CLR 300.

85 *Data Access Corporation v Powerflex Services Pty Ltd* [1999] HCA 49 at [84].

86 (1993) 176 CLR 300 at 330.

87 *Data Access Corporation v Powerflex Services Pty Ltd* [1999] HCA 49 at [86].

130 Properly understood, the *Autodesk* cases provide no support for the appellant's submissions with respect to the collocation of Reserved Words. However, that is not fatal to the argument that that collocation constitutes a substantial part of that program. What is fatal is that, as with the individual Reserved Words, the collocation is not part of a computer program at all. Neither individually nor in collocation are the Reserved Words part of the set of instructions that is intended to cause a computer to perform a particular function. And not being part, they cannot be a substantial part of it.

131 Orders should be made as proposed by Gleeson CJ, McHugh, Gummow and Hayne JJ.