

ALGOL

Sara Smith & Dave Stacy

CS 1621

September 15, 2003

History

In the 1950's and 1960's computer work was of mainly two kinds – commercial work and scientific calculations. Commercial work, or “data-processing, usually involves the storage and retrieval of large quantities of information but not a great deal of mathematical manipulation” (Ractliffe viii). On the other hand, scientific and engineering work requires “more sophisticated mathematical treatment . . . but less information storage” (Ractliffe viii). ALGOL is a scientific language.

“Computers were developed independently in many different institutions, firms and countries” with each computer model requiring its own language to suit its specific purposes (Ractliffe viii).

“In response to this language proliferation, several major computer-user groups in the United States, including SHARE (the IBM scientific user group) and USE (UNIVAC Scientific Exchange, the large-scale UNIVAC scientific user group), submitted a petition to the Association of Computing Machinery (ACM) on May 10, 1957, to form a committee to study and recommend action to create a universal programming language” (Sebesta 55).

“Previously, in 1955, GAMM (a German acronym for Society for Applied Mathematics and Mechanics) had also formed a committee to design one universal, machine-independent, algorithmic language for use on all kinds of computer” (Sebesta 55).

Fortran seemed like a good candidate however at the time it was solely owned by IBM

and so could not become a universal language (Sebesta 55). Each group sent four members to a meeting in Zurich from May 27 to June 1, 1958 (Sebesta 56).

They set forth three goals for the new language.

“The syntax . . . should be as close to possible to standard mathematical notation, and programs written in it should be readable with little further explanation. It should be possible to use [it] for the description of computing processes in publications. Programs . . . must be mechanically translatable into machine language” (Sebesta 56).

And thus, ALGOL 58 was born. The language was originally named International Algorithmic Language (IAL). However, the name was changed the following year to ALGOL (for ALGORithmic Language), a name that had been rejected during design since it did not reflect the international scope of the project.

ALGOL 58 was met enthusiastically in the United States at first. Several groups developed languages with ALGOL 58 as their basis. The University of Michigan gave birth to MAD, the United States Naval Electronics Group produced NELIAC and System Development Corporation designed and implemented JOVIAL which became the official scientific language of the US Air Force for a quarter of a century (Sebesta 57). IBM and SHARE both considered it for the 700 series computers and recommended that the ACM standardize it, however eventually decided to stick with Fortran as opposed to the pain and expense of getting a new language started (Sebesta 58).

While ALGOL was popular at first, it was debated endlessly and many additions and modifications were published. “Thirteen representatives, from Denmark, England, France, Germany, Holland, Switzerland, and the United States, conferred in Paris from January 11 to 16, 1960” (Backus 136) to revise the language. And so, ALGOL 60 was born. “Three different levels of language are recognized . . . a Reference Language, a Publication Language, and several Hardware Representations” (Backus 137).

The Reference Language is the “working language of the committee” and is the “basic reference and guide for compiler builders” and hardware representations. “The characters are determined by ease of understanding, not by computer limitations, coders notation or pure mathematical notation.” The Publication Language is used for “stating and communicating processes” and the “characters may be different in different countries.” Each of the Hardware Representations is specific to the particular computer, follow the rules set by the Reference Language (Backus 138).

Strengths

ALGOL 58 was a descendant of Fortran, though it “generalized many of Fortran’s features and added several constructs and concepts” in order to make the language more machine-independent, flexible and powerful (Sebesta 56). For instance, identifiers were allowed to have any length while Fortran I restricted them to six or fewer characters (Sebesta 57). ALGOL allowed any number of array dimensions and Fortran I allowed no more than three (Sebesta 57). Also, the array lower bound could be specified by the

programmer where Fortran I set it implicitly at 1 (Sebesta 57). ALGOL also allows nested selection statements while Fortran I does not (Sebesta 57).

Some new additions in ALGOL 58 included the formalization of “the concept of data type, although only variables that were not floating-point required explicit declaration” (Sebesta 57). It also “added the idea of compound statements” (Sebesta 57).

ALGOL 60 introduced block structure allowing a “programmer to localize parts of programs by introducing new data environments, or scopes” (Sebesta 58). It also allowed “two different means of passing parameters to subprograms – pass by value and pass by name” (Sebesta 58). Recursion was allowed for procedures (Sebesta 58) and stack dynamic arrays were introduced. In stack dynamic arrays subscript ranges specified by variables so size of array set execution time (Sebesta 59).

Weaknesses

Some features made ALGOL too flexible, difficult to understand and inefficient to implement. For instance, the pass-by-name method of passing parameters to subprograms was difficult to understand (Sebesta 60). Also, ALGOL lacks input and output statements. “Implementation-dependent input/output made programs difficult to port to other computers.” (Sebesta 60) BNF, Backus’s 1959 notation for describing programming language syntax, which was implemented in ALGOL 60, seemed “strange and complicated” in 1960 though it is now considered “simple and elegant” (Sebesta 60). Lastly, and perhaps most important to ALGOL’s failure was the lack of support from IBM and other Fortran users.

Usages

Among the many problem-oriented “programming languages, ALGOL holds a special position because of its international status and still more because of its strict and logical structure” (Studentlitteratur). Because of this, ALGOL is, or was, well suited for educational purposes, especially for those non-specialists, especially in Europe where it “obtained a strong position at most universities and technical institutes” (Studentlitteratur). It is useful in applications “with a background of mathematics and natural sciences . . . medical and social sciences” (Studentlitteratur).

ALGOL also became the “only acceptable formal means of communicating algorithms in computing literature” (Sebesta 59). It “remained for over twenty years the sole language for publishing algorithms” (Sebesta 59). It has many direct or indirect descendents including PL/I, SIMULA 67, ALGOL 68, C, Pascal, Ada, C++ and Java.

Sample Programs

Here are three sample ALGOL programs and the equivalent Java programs. Since ALGOL does not specify input or output system, manufacturers must make minor modifications to the Reference Language (Ractliffe ix), the print statements used here are consistent with the ALGOL 60 outlined in Ekman (see Bibliography).

Program 1 ALGOL (modified from Ekman 17)

```
begin  $T := 0.02$ ;  $x := 7$ ;  $y := 5$ ;  $t := 3$ ;  $pi := 3.14159265$ ;  
     $omega := 2 \times pi/T$ ;  
    begin  $z := x \times \cos(omega \times t) + y \times \sin(omega \times t)$ ;  
         $p := \text{sqrt}(z \uparrow 2 + 1)$   
    end;  
     $A := p \times T$ ;  $t := t+1$   
     $\text{print}(z)$ ;  
     $\text{print}(p)$ ;  
     $\text{print}(A)$ ;  
     $\text{print}(t)$ ;  
end
```

Program 1 Java

```
public class Program1Java  
{  
    public static void main(String[] args)  
    {  
        double T = 0.02;  
        double x = 7;  
        double y = 5;  
        double t = 3;  
        double omega = 2*Math.PI/T;  
        double z = (x * Math.cos(omega * t)) + (y * Math.sin(omega*t));  
        double p = Math.sqrt(Math.pow(z, 2) + 1);  
        double A = p*T;  
        t = t+1;  
        System.out.println(z); System.out.println(p);  
        System.out.println(A); System.out.println(t);  
    }  
}
```

Program 2 ALGOL (modified from Ekman 32)

```
begin integer s1, s2; real mean;  
    integer array A[1:4];  
    A[1] := 4; A[2] := 5; A[3] := 6; A[4] := 7;  
    s1 := A[1] + A[2] + A[3] + A[4]; mean := s1/4;  
    s2 := A[1] 2 + A[2] 2 + A[3] 2 + A[4] 2;  
    print(s1); print(mean); print(s2)  
end
```

Program 2 Java

```
public class Program2Java  
{  
    public static void main(String[] args)  
    {  
        int s1, s2;  
        double mean;  
        int[] A = new int[4];  
        A[0] = 4;  
        A[1] = 5;  
        A[2] = 6;  
        A[3] = 7;  
        s1 = A[0] + A[1] + A[2] + A[3];  
        mean = (double)s1/4;  
        s2 = A[0]*A[0] + A[1]*A[1] + A[2]*A[2] + A[3]*A[3];  
        System.out.println(s1);  
        System.out.println(mean);  
        System.out.println(s2);  
    }  
}
```

Program 3 ALGOL (Ekman 44)

```
begin real S; integer i, n;  
    S := 0;  
    n := 1; i := 10;  
L1:   S := S + 1/n/n;  
        if S ≥ 1.6449 then go to L2;  
        if n = i then begin print(n); print(S); i := i × 10 end;  
        n := n + 1;  
        go to L1;  
L2:   print(n); print(S)  
end
```

Program 3 Java

```
public class Program3Java  
{  
    public static void main(String[] args)  
    {  
        double S;  
        int i, n;  
        S = 0;  
        n = 1;  
        i = 10;  
        S = S + 1/(double)n/(double)n;  
        System.out.println(S);  
        while(S < 1.6449)  
        {  
            if(n == i)  
            {  
                System.out.println(n);  
                System.out.println(S);  
                i = i * 10;  
            }  
            n++;  
            S = S + 1/(double)n/(double)n;  
        }  
        System.out.println(n);  
        System.out.println(S);  
    }  
}
```

Bibliography

- Backus, J.W., et al. "Revised Report on the Algorithmic Language ALGOL 60."
Appendix. *Introduction to ALGOL Programming*. By Torgil Ekman and Carl-Erik Fröberg. London: Oxford University Press, 1967. 131-169.
- Ekman, Torgil and Carl-Erik Fröberg. *Introduction to ALGOL Programming*. London: Oxford University Press, 1967.
- Ractliffe, J. F. Introduction. *ALGOL in Brief: A Short, Practical Guide to Computer Programming in ALGOL*. By Ractliffe. London: Oxford University Press, 1971. vii-x.
- Sebesta, Robert W. *Concepts of Programming Languages*. Sixth Edition. Boston: Addison-Wesley, Pearson Education, Inc. 2004.
- Studentlitteratur, Lund, Sweden. Preface. *Introduction to ALGOL Programming*. By Torgil Ekman and Carl-Erik Fröberg. London: Oxford University Press, 1967.