

Hollerith constant

1 language

[Article](#) [Talk](#)

[Read](#) [Edit](#) [View history](#)

From Wikipedia, the free encyclopedia

Hollerith constants, named in honor of [Herman Hollerith](#), were used in early [FORTRAN](#) programs to allow manipulation of character data.

Early FORTRAN had no CHARACTER [data type](#), only numeric types. In order to perform character manipulation, characters needed to be placed into numeric variables using Hollerith constants. For example, the constant 3HABC specified a three-character string "ABC", identified by the initial integer representing the string length 3 and the specified Hollerith character H, followed by the string data ABC. These constants were *typeless*, so that there were no [type conversion](#) issues. If the constant specified fewer characters than was possible to hold in a data item, the characters were then stored in the item *left-justified* and *blank-filled*.

Mechanics [[edit](#)]

By the [FORTRAN 66](#) Standard, Hollerith syntax was allowed in the following uses:

- As constants in DATA statements
- As constant actual arguments in subroutine CALL statements
- As edit descriptors in FORMAT statements

Portability was problematic with Hollerith constants. First, [word](#) sizes varied on different computer systems, so the number of characters that could be placed in each data item likewise varied. Implementations varied from as few as two to as many as ten characters per word. Second, it was difficult to manipulate individual characters within a word in a portable fashion. This led to a great deal of *shifting and masking* code using non-standard, vendor-specific, features. The fact that character sets varied between machines also complicated the issue.

Some authors were of the opinion that for best portability, only a single character should be used per data item. However considering the small memory sizes of machines of the day, this technique was considered extremely wasteful.

Technological obsolescence [[edit](#)]

One of the major features of FORTRAN 77 was the CHARACTER string data type. Use of this data type dramatically simplified character manipulation in Fortran programs – rendering almost all uses of the Hollerith constant technique obsolete.

Hollerith constants were removed from the FORTRAN 77 Standard, though still described in an appendix for those wishing to continue support. Hollerith edit descriptors were allowed through Fortran 90, and were removed from the Fortran 95 Standard.

Examples [\[edit\]](#)

The following is a FORTRAN 66 [hello world](#) program using Hollerith constants. It assumes that at least four characters per word are supported by the implementation:

```
C      PROGRAM HELLO1
C
C      INTEGER IHWSTR(3)
C      DATA IHWSTR/4HHELL,4HO WO,3HRLD/
C
C      WRITE (6,100) IHWSTR
C      STOP
100    FORMAT (3A4)
C      END
```

Besides `DATA` statements, Hollerith constants were also allowed as actual arguments in subroutine calls. However, there was no way that the callee could know how many characters were passed in. The programmer had to pass the information explicitly. The [hello world](#) program could be written as follows – on a machine where four characters are stored in a word:

```
C      PROGRAM HELLO2
C      CALL WRTOUT (11HHELLO WORLD, 11)
C      STOP
C      END
C
C      SUBROUTINE WRTOUT (IARRAY, NCHRS)
C
C      INTEGER IARRAY(1) \[notes 1\]
C      INTEGER NCHRS
C
C      INTEGER ICPW
C      DATA ICPW/4/ \[notes 2\]
C      INTEGER I, NWRDS
C
C      NWRDS = (NCHRS + ICPW - 1) / ICPW
C      WRITE (6,100) (IARRAY(I), I=1,NWRDS)
C      RETURN
100    FORMAT (100A4) \[notes 3\]
C      END
```

Although technically not a Hollerith constant, the same Hollerith syntax was allowed as an *edit descriptor* in `FORMAT` statements. The [hello world](#) program could also be written as:

```
PROGRAM HELLO3
```

```
WRITE (6,100)
```

```
STOP
```

```
100 FORMAT (11HELLO WORLD)
```

```
END
```

One of the most surprising features was the behaviour of Hollerith edit descriptors when used for input. The following program would change at run time `HELLO WORLD` to whatever would happen to be the next eleven characters in the input stream and print that input:

```
PROGRAM WHAT1
```

```
READ (5,100)
```

```
WRITE (6,100)
```

```
STOP
```

```
100 FORMAT (11HELLO WORLD)
```

```
END
```

Notes [\[edit \]](#)

- [^] FORTRAN 66 did not have a way to indicate a variable-sized array. So a '1' was typically used to indicate that the size is unknown.
- [^] Four characters per word.
- [^] A count of 100 is a 'large enough' value that any reasonable number of characters can be written. Also note that four characters per word is hard-coded here too.

References [\[edit \]](#)

- American Standard FORTRAN*. American Standards Association, X3.9-1966. p. 38.

Categories: [Fortran](#) | [String data structures](#)

This page was last edited on 23 April 2026, at 15:22 (UTC). Page was rendered with [Parsoid](#).

Text is available under the [Creative Commons Attribution-ShareAlike 4.0 License](#); additional terms may apply. By using this site, you agree to the [Terms of Use](#) and [Privacy Policy](#). Wikipedia® is a registered trademark of the [Wikimedia Foundation, Inc.](#), a non-profit organization.

[Privacy policy](#) [About Wikipedia](#) [Disclaimers](#) [Contact Wikipedia](#) [Legal & safety contacts](#) [Code of Conduct](#) [Developers](#) [Statistics](#)

[Cookie statement](#) [Mobile view](#)

