In software development, **obfuscation** is the deliberate act of creating source or machine code that is difficult for humans to understand. Like obfuscation in natural language, it may use needlessly roundabout expressions to compose statements. Programmers may deliberately obfuscate code to conceal its purpose (security through obscurity) or its logic or implicit values embedded in it, primarily, in order to prevent tampering, deter reverse engineering, or even as a puzzle or recreational challenge for someone reading the source code. This can be done manually or by using an automated tool, the latter being the preferred technique in industry.[1]

## Overview

The architecture and characteristics of some languages may make them easier to obfuscate than others.[2][3] C,[4] C++,[5][6] and the Perl programming language[7] are some examples of languages easy to obfuscate.

## Recreational obfuscation

Writing and reading obfuscated source code can be a brain teaser. A number of programming contests reward the most creatively obfuscated code, such as the International Obfuscated C Code Contest and the Obfuscated Perl Contest.

Types of obfuscations include simple keyword substitution, use or non-use of whitespace to create artistic effects, and self-generating or heavily compressed programs.

According to Nick Montfort, techniques may include:

1. naming obfuscation, which includes naming variables in a meaningless or deceptive way;
2. data/code/comment confusion, which includes making some actual code look like comments or confusing syntax with data;
3. double coding, which can be displaying code in poetry form or interesting shapes.[8]

Short obfuscated Perl programs may be used in signatures of Perl programmers. These are JAPHs ("Just another Perl hacker").[9]

### Examples

This is a winning entry from the International Obfuscated C Code Contest written by Ian Phillipps in 1988[10] and subsequently reverse engineered by Thomas Ball.[11]

```
/*
  LEAST LIKELY TO COMPILE SUCCESSFULLY:
  Ian Phillipps, Cambridge Consultants Ltd., Cambridge, England
*/

#include <stdio.h>
main(t,_,a)
char
*
a;
{
    return!

0<t?
t<3?

main(-79,-13,a+
main(-87,1-_,
main(-86, 0, a+1 )

+a)):

1,
t<_?
main(t+1, _, a )
:3,

main ( -94, -27+t, a )
&&t == 2 ?_
<13 ?

main ( 2, _+1, "%s %d %d\n" )

:9:16:
t<0?
t<-72?
main( _, t,
"@n'+,#'/*{}w+/w#cdnr/+,{}r/*de}+,/*{*+,/w{%+,/w#q#n+,/#
{l,+,/n{n+,/+#n+,/#;\
```

```
#q#n+,/+k#;*+,/'r :'d*'3,}{w+K w'K:'+}e#';dq#'l q#'+d'K#!/+k#;\
q#'r}eKK#}w'r}eKK{nl]'/#;#q#n'){)#}w'){){nl]'/+#n';d}rw' i;# ){nl]!/n{n#';
\
r{#w'r nc{nl]'/#{l,+'K {rw' iK{;[{nl]'/w#q#\
\
n'wk nw' iwk{KK{nl]!/w{%'l##w#' i; :{nl]'/*{q#'ld;r'}{nlwb!/*de}'c ;;\
{nl'-{}rw]'/+,}##'*}#nc,',#nw]'/+kd'+e}+;\
#'rdq#w! nr'/ ') }+}{rl#'{n' ')# }'+}##(!!/")
:
t<-50?
_==*a ?
putchar(31[a]):

main(-65,_,a+1)
:
main((*a == '/') + t, _, a + 1 )
:

0<t?

main ( 2, 2 , "%s")
:*a=='/'||

main(0,

main(-61,*a, "!ek;dc i@bK'(q)-[w]*%n+r3#l,{}:\nuwloca-O;m
.vpbks,fxntdCeghiry")

,a+1);}
```

It is a C program that when compiled and run will generate the 12 verses of *The 12 Days of Christmas*. It contains all the strings required for the poem in an encoded form within the code.

A non-winning entry from the same year, this next example illustrates creative use of whitespace; it generates mazes of arbitrary length:[12]

```
char*M,A,Z,E=40,J[40],T[40];main(C){for(*J=A=scanf(M="%d",&C);
--           E;           J[           E]           =T
[E   ]=  E)   printf("._");  for(;(A-=Z=!Z)  ||  (printf("\n|"
)    ,   A    =            39           ,C           --
```

```
)    ;   Z    ||    printf   (M   ))M[Z]=Z[A-(E   =A[J-Z])&&!C
&    A   ==             T[                                A]
|6<<27<rand()||!C&!Z?J[T[E]=T[A]]=E,J[T[A]=A-Z]=A,"_.":" |"];}
```

ANSI-compliant C compilers don't allow constant strings to be overwritten, which can be avoided by changing "*M" to "M[3]" and omitting "M=".

The following example by Óscar Toledo Gutiérrez, Best of Show entry in the 19th IOCCC, implements an 8080 emulator complete with terminal and disk controller, capable of booting CP/M-80 and running CP/M applications:[13]

```
#include <stdio.h>
          #define n(o,p,e)=y=(z=a(e)%16 p x%16 p o,a(e)p x p o),h(
                              #define s 6[o]
            #define p z=l[d(9)]|l[d(9)+1]<<8,1<(9[o]+=2)||++8[o]
                              #define Q a(7)
          #define w 254>(9[o]-=2)||--8[o],l[d(9)]=z,l[1+d(9)]=z>>8
                             #define O )):((
                  #define b (y&1?~s:s)>>"\6\0\2\7"[y/2]&1?0:(
                              #define S )?(z-=
                    #define a(f)*((7&f)-6?&o[f&7]:&l[d(5)])
                              #define C S 5 S 3
                       #define D(E)x/8!=16+E&198+E*8!=x?
                             #define B(C)fclose((C))
                       #define q (c+=2,0[c-2]|1[c-2]<<8)
                          #define m x=64&x?*c++:a(x),
                         #define A(F)=fopen((F),"rb+")
                    unsigned char o[10],l[78114],*c=l,*k=l
                          #define d(e)o[e]+256*o[e-1]
#define h(l)s=l>>8&1|128&y|!
(y&255)*64|16&z|2,y^=y>>4,y^=y<<2,y^=~y>>1,s|=y&4
+64506; e,V,v,u,x,y,z,Z; main(r,U)char**U;{

     { { { } } }       { { { } } }       { { { } } }       { { { } } }
    { { {   } } }     { { {   } } }     { { {   } } }     { { {   } } }
   { { {     } } }   { { {     } } }   { { {     } } }   { { {     } } }
   { { {     } } }   { { {     } } }   { { {     } } }   { { {     } } }
   { { {     } } }   { { {     } } }   { { {     } } }   { { {     } } }
    { { {   } } }     { { {     } } }     { { {   } } }   { { {     } } }
      { { ; } }       { { {     } } }       { { ; } }     { { {     } } }
```

```
    { { {   } } }    { { {     } } }    { { {   } } }    { { {     } } }
   { { {     } } }   { { {     } } }   { { {     } } }   { { {     } } }
   { { {     } } }   { { {     } } }   { { {     } } }   { { {     } } }
   { { {     } } }   { { {     } } }   { { {     } } }   { { {     } } }
    { { {   } } }     { { {   } } }     { { {   } } }     { { {   } } }
     { { { } } }       { { { } } }       { { { } } }       { { { } } }
```

```
                                    for(v A((u A((e A((r-2?0:(V
A(1[U])),"C")
),system("stty raw -echo min 0"),fread(l,78114,1,e),B(e),"B")),"A")); 118-
(x
=*c++); (y=x/8%8,z=(x&199)-4 S 1 S 1 S 186 S 2 S 2 S 3 S 0,r=(y>5)*2+y,z=
(x&
207)-1 S 2 S 6 S 2 S 182 S 4)?D(0)D(1)D(2)D(3)D(4)D(5)D(6)D(7)(z=x-2 C C C
C
C C C C+129 S 6 S 4 S 6 S 8 S 8 S 6 S 2 S 2 S 12)?x/64-1?((0 O a(y)=a(x) O
9
[o]=a(5),8[o]=a(4) O 237==*c++?((int (*)())(2-*c++?fwrite:fread))
(l+*k+1[k]*
256,128,1,(fseek(y=5[k]-1?u:v,((3[k]|4[k]<<8)<<7|2[k])<<7,Q=0),y)):0 O
y=a(5
),z=a(4),a(5)=a(3),a(4)=a(2),a(3)=y,a(2)=z O c=l+d(5) O
y=l[x=d(9)],z=l[++x]
,x[l]=a(4),l[--x]=a(5),a(5)=y,a(4)=z O 2-*c?Z||read(0,&Z,1),1&*c++?
Q=Z,Z=0:(
Q=!!Z):(c++,Q=r=V?fgetc(V):-1,s=s&~1|r<0) O++c,write(1,&7[o],1) O z=c+2-
l,w,
c=l+q O p,c=l+z O c=l+q O s^=1 O Q=q[l] O s|=1 O q[l]=Q O Q=~Q O
a(5)=l[x=q]
,a(4)=l[++x] O s|=s&16|9<Q%16?Q+=6,16:0,z=s|=1&s|Q>159?
Q+=96,1:0,y=Q,h(s<<8)
O l[x=q]=a(5),l[++x]=a(4) O x=Q%2,Q=Q/2+s%2*128,s=s&~1|x O Q=l[d(3)]O x=Q
/
128,Q=Q*2+s%2,s=s&~1|x O l[d(3)]=Q O s=s&~1|1&Q,Q=Q/2|Q<<7 O Q=l[d(1)]O
s=~1
&s|Q>>7,Q=Q*2|Q>>7 O l[d(1)]=Q O m y n(0,-,7)y) O m z=0,y=Q|=x,h(y) O m
z=0,
y=Q^=x,h(y) O m z=Q*2|2*x,y=Q&=x,h(y) O m Q n(s%2,-,7)y) O m Q n(0,-,7)y)
O
m Q n(s%2,+,7)y) O m Q n(0,+,7)y) O z=r-8?d(r+1):s|Q<<8,w O p,r-8?
```

```
o[r+1]=z,r
[o]=z>>8:(s=~40&z|2,Q=z>>8) O r[o]--||--o[r-1]O
a(5)=z=a(5)+r[o],a(4)=z=a(4)
+o[r-1]+z/256,s=~1&s|z>>8 O ++o[r+1]||r[o]++O o[r+1]=*c++,r[o]=*c++O z=c-
l,w
,c=y*8+l O x=q,b z=c-l,w,c=l+x) O x=q,b c=l+x) O b p,c=l+z) O a(y)=*c++O
r=y
,x=0,a(r)n(1,-,y)s<<8) O r=y,x=0,a(r)n(1,+,y)s<<8))));
system("stty cooked echo"); B((B((V?B(V):0,u)),v)); }
```

An example of a JAPH:

```
@P=split//,".URRUU\c8R";@d=split//,"\nrekcah xinU / lreP rehtona tsuJ";sub
p{
@p{"r$p","u$p"}=(P,P);pipe"r$p","u$p";++$p;
($q*=2)+=$f=!fork;map{$P=$P[$f^ord
($p{$_})&6];$p{$_}=/ ^$P/ix?
$P:close$_}keys%p}p;p;p;p;p;map{$p{$_}=~/^[P.]/&&
close$_}%p;wait until$?;map{/^r/&&<$_>}%p;$_=$d[$q];sleep
rand(2)if/\S/;print
```

This slowly displays the text "Just another Perl / Unix hacker", multiple characters at a time, with delays. An explanation can be found here.[14]

Some Python examples can be found in the official Python programming FAQ and elsewhere.[15][16][17]

## Advantages of obfuscation

There are several advantages of automated code obfuscation that have made it popular and widely useful across many platforms. On some platforms (such as Java,[18] Android,[19] and .NET) a decompiler can reverse-engineer source code from an executable or library. A main advantage of automated code obfuscation is that it helps protect the trade secrets (intellectual property) contained within software by making reverse-engineering a program difficult and economically unfeasible. Other advantages might include helping to protect licensing mechanisms and unauthorized access, and shrinking the size of the source code, and possibly shrinking the size of the executable. Decompilation is sometimes called a man-at-the-end attack, based on the traditional cryptographic attack known as "man-in-the-middle". For run-time interpreted languages (more commonly known as script), like older versions of BASIC, programs execute

faster and take less RAM if they use single letter variable names, avoid comments and do not contain blank characters. (in brief, the shorter the faster)

## Disadvantages of obfuscation

While obfuscation can make reading, writing, and reverse-engineering a program difficult and time-consuming, it will not necessarily make it impossible.[20] Some anti-virus software, such as AVG AntiVirus, will also alert their users when they land on a site with code that is manually obfuscated, as one of the purposes of obfuscation can be to hide malicious code. However, some developers may employ code obfuscation for the purpose of reducing file size or increasing security. The average user may not expect their antivirus software to provide alerts about an otherwise harmless piece of code, especially from trusted corporations, so such a feature may actually deter users from using legitimate software.

Certain major browsers such as Firefox and Chrome also disallow browser extensions containing obfuscated code. [21][22]

## Obfuscating software

A variety of tools exist to perform or assist with code obfuscation. These include experimental research tools created by academics, hobbyist tools, commercial products written by professionals, and open-source software. There also exist deobfuscation tools that attempt to perform the reverse transformation.

Although the majority of commercial obfuscation solutions work by transforming either program source code,[23][24] or platform-independent bytecode as used by Java[25] and .NET,[26] there are also some that work directly on compiled binaries.

## Obfuscation and copyleft licenses

There has been debate on whether it is illegal to skirt copyleft software licenses by releasing source code in obfuscated form, such as in cases in which the author is less willing to make the source code available. The issue is addressed in the GNU General Public License by requiring the "preferred form for making modifications" to be made available.[27] The GNU website states "Obfuscated 'source code' is not real source code and does not count as source code." [28]

## See also

- AARD code

- Spaghetti code
- Write-only language
- Decompilation
- Esoteric programming language
- Quine
- Polymorphic code
- Hardware obfuscation
- Underhanded C Contest
- Source-to-source compiler
- ProGuard (Java Obfuscator)
- Dotfuscator (.Net Obfuscator)
- Digital rights management
- Indistinguishability obfuscation

## Notes


1. "What is obfuscation (obfu)? - Definition from WhatIs.com" . *SearchSoftwareQuality*. Retrieved February 1, 2019.
2. Binstock, Andrew (March 6, 2003). "Obfuscation: Cloaking your Code from Prying Eyes" . Web.archive.org. Archived from the original on April 20, 2008. Retrieved November 25, 2013.
3. Atwood, Jeff (May 15, 2005). "Jeff Atwood, May 15, 2005" . Codinghorror.com. Retrieved November 25, 2013.
4. "Obfuscation" . Kenter.demon.nl. Retrieved November 25, 2013.
5. "C++ Tutorials - Obfuscated Code - A Simple Introduction" . DreamInCode.net. Retrieved November 25, 2013.
6. "C Tutorials - Obfuscated Code in C" . Sites.google.com. July 7, 2011. Retrieved November 25, 2013.
7. As of 2013-11-25 18:22 GMT. "Pe(a)rls in line noise" . Perlmonks.org. Retrieved November 25, 2013.
8. Montfort, Nick. "Obfuscated code" (PDF). Retrieved November 24, 2017.

9. "JAPH - Just Another Perl Hacker" . *pm.org*. Perl Mongers. Archived from the original on May 16, 2013. Retrieved February 27, 2015.
10. "International Obfuscated C Code Winners 1988 - Least likely to compile successfully" . Ioccc.org. Retrieved November 25, 2013.
11. " "Reverse Engineering the Twelve Days of Christmas" by Thomas Ball" . Research.microsoft.com. Archived from the original on December 13, 2007. Retrieved November 25, 2013.
12. Don Libes, *Obfuscated C and Other Mysteries*, John Wiley & Sons, 1993, pp 425. ISBN 0-471-57805-3
13. Óscar Toledo Gutiérrez: *Intel 8080 emulator. 19th IOCCC. Best of Show.*
14. "Obfuscated Perl Program" . Perl.plover.com. Retrieved November 25, 2013.
15. "Obfuscating "Hello world!" – Ben Kurtovic" . *benkurtovic.com*.
16. http://wiki.c2.com/?ObfuscatedPython
17. https://code.activestate.com/lists/python-list/16171/ "The First Annual Obfuscated Python Content"
18. " "Decompiling Java" by Godfrey Nolan" . Apress;.
19. " "Decompiling Android" by Godfrey Nolan" . Apress;.
20. " "Can We Obfuscate Programs?" by Boaz Barak" . Math.ias.edu. Retrieved November 25, 2013.
21. at 05:01, Thomas Claburn in San Francisco 2 Oct 2018. "Google taking action against disguised code in Chrome Web Store" . *www.theregister.co.uk*. Retrieved November 12, 2019.
22. Cimpanu, Catalin. "Mozilla announces ban on Firefox extensions containing obfuscated code" . *ZDNet*. Retrieved November 12, 2019.
23. "Open Directory - Computers: Programming: Languages: JavaScript: Tools: Obfuscators" . Dmoz.org. August 3, 2013. Retrieved November 25, 2013.
24. "Open Directory - Computers: Programming: Languages: PHP: Development Tools: Obfuscation and Encryption" . Dmoz.org. September 19, 2013. Retrieved November 25, 2013.
25. "Open Directory - Computers: Programming: Languages: Java: Development Tools: Obfuscators" . Dmoz.org. April 9, 2013. Retrieved November 25, 2013.
26. "Open Directory - Computers: Programming: Component Frameworks: .NET: Tools: Obfuscators" . Dmoz.org. January 2, 2007. Retrieved November 25, 2013.

27. "Reasoning behind the "preferred form of the work for making modifications to it" language in the GPL" . Lwn.net. Retrieved November 25, 2013.

28. "What is free software?" . gnu.net. Retrieved December 18, 2014.

## References

- Seyyedhamzeh, Javad, ABCME: A Novel Metamorphic Engine , 17th National Computer Conference, Sharif University of Technology, Tehran, Iran, 2012.
- B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. Vadhan and K. Yang. "On the (Im)possibility of Obfuscating Programs" . *21st Annual International Cryptology Conference*, Santa Barbara, California, USA. Springer Verlag LNCS Volume 2139, 2001.
- Mateas, Michael; Nick Montfort. "A Box, Darkly: Obfuscation, Weird Languages, and Code Aesthetics" (PDF). *Proceedings of the 6th Digital Arts and Culture Conference, IT University of Copenhagen, 1–3 December 2005*. pp. 144–153.

## External links

- The International Obfuscated C Code Contest
- Protecting Java Code Via Code Obfuscation , ACM Crossroads, Spring 1998 issue
- Protect Your Java Code - Through Obfuscators And Beyond , April 2009
- Dotfuscator in — Visual Studio documentation for built-in .NET obfuscation
- Obfuscation tools for .NET, on MSDN — Obfuscation resources for .NET, on the Microsoft Developer Center.
- Can we obfuscate programs?
- Yury Lifshits. Lecture Notes on Program Obfuscation (Spring'2005)
- Java obfuscators at Curlie
- Analysis of the 12 days program
- Analysis of the obfuscated maze generating program
- Obfuscated Perl program with explanation
- Analysis of javascript code obfuscation
- c2:BlackBoxComputation