Home * Programming * Languages * **Pascal**

Blaise Pascal [1]

**Pascal** is an imperative, procedural programming language developed and published in 1970 by Niklaus Wirth from ETH Zurich. It is a small and efficient language intended to encourage good programming practices using structured programming and data structuring. Pascal is based on Algol and named in honor of the French mathematician and philosopher Blaise Pascal, like many programming languages, but unlike C, Pascal allows nested procedure definitions to any level of depth, and also allows most kinds of definitions and declarations inside procedures and functions.

# Table of Contents

UCSD Pascal

Turbo Pascal

Inline Assembly

Source Sample

See also

Pascal Engines

Publications

Forum Posts

External Links

References

What links here?

# UCSD Pascal

A notable Pascal system was UCSD-Pascal from University of California, San Diego that ran on a machine-independent operating system, the UCSD p-System portable. In the early 1980s, UCSD Pascal was ported to Apple II computers to provide a structured alternative to the Basic interpreters that came with the machines.

# Turbo Pascal

An increased popularity of Pascal on home computers of the early 80s arose from the fast one-pass Turbo Pascal compiler, running under CP/M for 8-bit 8080/Z80 based computers, as well as CP/M-86 and MS-DOS for 8086 and later x86 based 16- and 32-bit computers. Borland licensed the PolyPascal compiler core [2], written by Anders Hejlsberg, and added user interface and editor, considered as one of the first Integrated development environment for home computers. Anders Hejlsberg later joined the company as an employee and was the architect for all versions of the Turbo Pascal compiler and the first three versions of Borland Delphi.

By 1995 Borland had dropped Turbo Pascal and replaced it with the RAD environment Delphi, based on Object Pascal. Early versions of Turbo Pascal are free downloadable from Embarcadero Technologies [3].

## Inline Assembly

Early versions of Turbo Pascal could include inline machine code as hexadecimal Byte sequence, while later Inline Assembly was suported. None portable Inline assembly enabled programmers to access hardware features that were otherwise not directly available, and it was in fact overused by experienced Turbo Pascal programmers, as it could be seen from the source code of Turbo Chess [4], available from the *Turbo GameWorks Book* [5] by Kaare Danielsen [6].

## Source Sample

A sample of a recursive Search routine in Turbo Pascal from KC Chess: Kevin & Craig's Chess Program [7]:

```
{----------------------------------------------------------------
```

```
-------}
{  Search:  This routine handles all of the tree searching except the
first  }
{  level which of the tree which is handled by the main routine.  Give
n the  }
{  player, all of his moves are generated, and then each one is made.
 The   }
{  enemy's maximum countermove score is subtracted from the move score
, and  }
{  this gives the net value for the player making the move.  The maxim
um    }
{  net score is remembered and returned by this function.  The player'
s move }
{  is then taken back, and all of his other possible moves are tried i
n this }
{  same way.  If the score of any move exceeds the given cutoff value,
 then  }
{  no other of the player's moves are checked, and the score that exce
eded   }
{  (or matched) the cutoff value is returned.  If the given depth is o
ne or  }
{  less, then the enemy's countermoves are not checked, only the point
s for  }
{  taking the pieces, minus the points of the player's piece if the en
emy's  }
{  piece is protected.  However, if the current player's move being
       }
{  considered is a take and it is given that all of moves to that poin
t      }
{  have been AllTakes, then enemy countermoves will be checked down to
 a    }
{  given depth of -1.  To calculate the enemy's best score in retaliat
ion    }
{  to the given player's move, this routine is called recursively with
 the   }
{  enemy as the player to move, and a depth of the one originally give
n, -1. }
{  The new cutoff value is the score of the move that was just made, m
inus   }
{  the the best score that was found so far.  If the enemy's countermo
ves    }
{  score exceeds or matches this countermove value, then the net score
 of    }
{  the original player's move cannot exceed the best found so far, and
 the   }
{  move will be thrown out.  The new value for AllTakes is passed on a
```

```
s true }
{  if all moves heretofor have been takes, and the current player's mo
ve is  }
{  a take.  This routine is the core of the computer's 'thinking'.
       }
{-------------------------------------------------------------------
-------}
  function Search (Turn: PieceColorType; CutOff, Depth: integer;
 AllTakes : boolean) : integer;
    var MoveList: MoveListType;
        j, LineScore, Score, BestScore, STCutOff: integer;
        Movement: MoveType;
        Attacked, Protected: integer;
        NoMoves, TakingPiece: boolean;
    begin
      {*** get the player's move list ***}
      GenMoveList (Turn, MoveList);
      NoMoves := true;
      BestScore := NegInfinity;
      j := MoveList.NumMoves;

      {*** go through all of the possible moves ***}
      while (j > 0) do begin
          Movement := MoveList.Move[j];
          {*** make the move ***}
          MakeMove (Movement, false, Score);
          {*** make sure it is legal (not moving into check) ***}
          AttackedBy (Player[Turn].KingRow, Player[Turn].
KingCol, Attacked, Protected);
          if (Attacked = 0) then begin
              NoMoves := false;
              if (Score = STALE_SCORE) then
                  {*** end the search on a stalemate ***}
                  LineScore := Score
              else begin
                  TakingPiece := Movement.PieceTaken.image <> BLANK;
                  if (Depth <= 1) and not (AllTakes and
 TakingPiece and (Depth >= PLUNGE_DEPTH)) then begin

{*** have reached horizon node of tree: score points for piece taken *
**}

{*** but assume own piece will be taken if enemy's piece is protected
***}
                      if Movement.PieceTaken.image <> BLANK then begin
                          Attacke
```

```
dBy (Movement.ToRow, Movement.ToCol, Attacked, Protected);
                          if Attacked > 0 then

LineScore := Score - CapturePoints[Movement.PieceMoved.image]
                          else
                              LineScore := Score;
                      end else
                          LineScore := Score;
                  end else begin
                      {*** new cutoff value ***}
                      STCutOff := Score - BestScore;

{*** recursive call for enemy's best countermoves score ***}
                      LineScore := Score - Search (
EnemyColor[Turn], STCutOff,


            Depth - 1, AllTakes and TakingPiece);
                  end;
              end;
              {*** remember player's maximum net score ***}
              if (LineScore > BestScore) then BestScore := LineScore;
          end;
          {*** un-do the move and check for cutoff ***}
          UnMakeMove (Movement);
          if BestScore >= CutOff then j := 0 else j := j - 1;
      end;
      if (BestScore = STALE_SCORE) then
          BestScore := -STALE_SCORE;
{stalemate means both players lose}
      if NoMoves then
          {*** player cannot move ***}
          if Player[Turn].InCheck then
              {*** if he is in check and cannot move, he loses ***}
              BestScore := - CapturePoints[KING]
          else
              {*** if he is not in check, then both players lose ***}
              BestScore := -STALE_SCORE;
{prefer stalemate to checkmate}
      Search := BestScore;
    end; {Search}
```

# See also

- Algol

- Delphi
- Modula-2

# Pascal Engines

Dynamic list with tag 'pascal'. Engines (at least some versions) written in Pascal:

1. Ananse
2. BBchess
3. Bestia
4. Betsy
5. BlackBishop
6. Bobby
7. Centaur
8. Chat
9. Chaturanga
10. Chess 0.5
11. CookieCat
12. Critter
13. Dappet
14. Delfi
15. Delphil
16. Delta
17. Diogenes
18. Eveann
19. Fafner
20. Goldbar
21. Gustav
22. HAL
23. Hector
24. Holmes
25. Ikarus
26. Joanna
27. Jupiter
28. KC Chess
29. Killer
30. Killer (NL)
31. LTChess
32. Matchess
33. Merlin
34. Nemeton
35. Neptune
36. Nero
37. Neurosis

38. Nexus
39. Now
40. Pachera
41. Paul
42. PawnKing
43. Popeye
44. Prédateur
45. Pschach
46. Rex
47. Storm
48. Turbo Chess
49. White Knight
50. XiniX
51. Ziggy
52. Zzzzzz

# Publications


- Brian W. Kernighan (**1981**). Why Pascal is Not My Favorite Programming Language. pdf
- Kaare Danielsen (**1985**). Turbo GameWorks. Borland International
- Don Beal (**1986**). *Turbo GameWorks: Tools for Turbo Pascal*. (Review) ICCA Journal, Vol. 9, No. 2, pp. 88
- Christopher Chabris (**1987**). *Artificial Intelligence and Turbo Pascal - Book and Disk*. Irwin Professional Pub, amazon.com » Artificial Intelligence


# Forum Posts

- Is it worth to program chess in Pascal ? (Re: My first Chess program !) by Ren Wu, rgcc, September 22, 1997
- Update for Byte Chess 0.5 by I Forget, comp.lang.pascal.misc, June 12, 2005 » Chess 0.5 [8] [9]
- Critter: Pascal vs C by Richard Vida, CCC, August 27, 2009 » C
- FreePascal 64 bit question by Dann Corbit, CCC, January 21, 2011
- for Pascal fans: Critter by Richard Vida, CCC, September 16, 2011
- Announcement: The Bozochess Project by Steven Edwards, CCC, October 05, 2011
- CookieCat Monday release schedule by Steven Edwards, CCC, December 19, 2011 » CookieCat
- CookieCat and perft by Steven Edwards, CCC, October 14, 2012
- New open source chess program by lauriet, OpenChess Forum, November 03, 2013 » LTChess

# External Links

- Pascal (programming language) from Wikipedia
- Pascal Users group Newsletter
- Niklaus Wirth's 2004/10/20 Lecture for The Computer History Museum

- ANSI-ISO PASCAL hosted by Scott Moore
- Embarcadero Delphi from Wikipedia
- Free Pascal from Wikipedia
- Free Pascal - Advanced open source Pascal compiler for Pascal and Object Pascal - Home Page
- GNU Pascal from Wikipedia
- GNU Pascal
- Lazarus (IDE) from Wikipedia
- Object Pascal from Wikipedia
- Turba Pascal from Wikipedia
- UCSD-Pascal from Wikipedia
- Virtual Pascal from Wikipedia
- comp.lang.pascal
- An Interview with Niklaus Wirth by Christian Timmerer with László Böszörményi, Part 2, YouTube Video

# References


1. ^ Blaise Pascal from Wikipedia
2. ^ Innovation Happens Elsewhere
3. ^ Embarcadero Developer Network - Museum
   Antique Software: Turbo Pascal v1.0
   Antique Software: Turbo Pascal v3.02
4. ^ Turbo Pascal - Assembly language
5. ^ Turbo GameWorks (Open Library)
6. ^ Resume for Kaare Danielsen
7. ^ KC CHESS: Kevin & Craig's Chess Program
8. ^ Chess 0.5, Release 1 - 2005-05-30
9. ^ Byte Chess 0.5 source code


# What links here?

| Page | Date Edited |
|---|---|
| 4th Computer Olympiad | Jul 15, 2017 |
| 8080 | Sep 3, 2017 |
| ACM 1981 | Jan 19, 2018 |
| ACM 1982 | Jul 19, 2016 |
| ACM 1984 | Jul 19, 2016 |
| ACM 1986 | Dec 5, 2017 |
| ACM 1990 | Jun 7, 2016 |
| ACM 1993 | Oct 30, 2017 |
| ACM 1994 | May 5, 2017 |
| Algol | Feb 1, 2014 |
| Amir Ban | Jan 24, 2016 |
| Array | Dec 1, 2016 |

| Page | Date Edited |
|---|---|
| Artificial Intelligence | Apr 9, 2018 |
| Assembly | Sep 3, 2017 |
| BBchess | Oct 30, 2011 |
| Bernard Brioit | Jan 10, 2015 |
| Bestia | Jan 5, 2013 |
| Betsy | Jul 28, 2014 |
| Bobby | Jan 7, 2016 |
| C | Feb 19, 2018 |
| C sharp | Nov 26, 2017 |
| CDC Cyber | Dec 22, 2017 |
| Chat | Jan 7, 2016 |
| Chaturanga | Apr 4, 2013 |
| Chess (Program) | Dec 22, 2017 |
| Chess 0.5 | Nov 20, 2016 |
| Chess 0.5X | Nov 5, 2015 |
| Chess Wizard | Dec 22, 2014 |
| Christopher Chabris | Aug 15, 2015 |
| CookieCat | Nov 15, 2016 |
| Craig S. Bruce | Apr 14, 2014 |
| Critter | Jan 25, 2014 |
| Dappet | Dec 12, 2016 |
| DarkThought | Jul 3, 2018 |
| David Podber | Jan 5, 2015 |
| Delfi | Jan 7, 2016 |
| Delphi | Dec 16, 2016 |
| Delphil | Sep 17, 2016 |
| Delta | May 22, 2016 |
| Diogenes | Jan 8, 2016 |
| DOCCC 1981 | Nov 14, 2015 |
| DOCCC 1982 | Nov 14, 2015 |
| DOCCC 1984 | May 22, 2016 |
| DOCCC 1985 | Dec 9, 2016 |
| DOCCC 1992 | Dec 12, 2016 |
| Engines | Mar 10, 2018 |
| Eugen | Jan 7, 2016 |
| Eveann | Apr 3, 2017 |
| Evgeniy Korniloff | Nov 22, 2015 |
| Fafner | Jan 8, 2016 |
| General Setwise Operations | Feb 25, 2018 |
| Goldbar | Jul 23, 2015 |
| Gustav | May 14, 2016 |
| HAL | Jun 14, 2016 |
| Hector | Nov 11, 2016 |
| iCE | Jan 4, 2016 |
| Ikarus | Jul 5, 2016 |

| Page | Date Edited |
|---|---|
| James Swafford | Jan 1, 2018 |
| Joanna | Dec 9, 2014 |
| John Stanback | Dec 7, 2016 |
| Julien Marcel | Oct 19, 2016 |
| Junior | Feb 27, 2018 |
| Jupiter | Sep 2, 2015 |
| Kaare Danielsen | Feb 19, 2017 |
| KC Chess | Jan 28, 2012 |
| Killer | Dec 15, 2017 |
| Killer (NL) | Dec 15, 2017 |
| Languages | Nov 26, 2017 |
| Larry Atkin | Jan 7, 2016 |
| Laurie Tunnicliffe | Jun 13, 2017 |
| LTChess | Mar 30, 2017 |
| Marcus Wagner | Jan 7, 2016 |
| Martin Bryant | Dec 11, 2016 |
| Matchess | Sep 7, 2015 |
| Mathematician | Apr 9, 2018 |
| Max | Dec 23, 2017 |
| Merlin | Jan 20, 2018 |
| Modula-2 | Aug 8, 2010 |
| MTD(f) | Jul 17, 2017 |
| Nemeton | Dec 23, 2017 |
| Neptune | Nov 11, 2016 |
| Nero | May 5, 2017 |
| Neurosis | Dec 23, 2017 |
| Nexus | Jan 7, 2016 |
| Norman Whaland | Dec 25, 2017 |
| Now | Jan 7, 2016 |
| Pachera | Nov 12, 2015 |
| Pascal | Nov 28, 2016 |
| Paul | Jan 8, 2016 |
| Paul Ramsteijn | Nov 13, 2015 |
| PawnKing | Sep 5, 2014 |
| Peter Gillgasch | May 15, 2016 |
| Peter W. Frey | Dec 25, 2017 |
| Philippe Fabiani | Sep 15, 2016 |
| Popeye | Oct 11, 2017 |
| Prédateur | Nov 11, 2016 |
| Pschach | Mar 22, 2016 |
| Pseudorandom number generator | May 11, 2017 |
| R. Kevin Phillips | Jan 28, 2012 |
| Ren Wu | Jul 23, 2017 |
| Rex | Jan 7, 2016 |
| RexChess | Dec 7, 2016 |

| Page | Date Edited |
| --- | --- |
| Richard Vida | Aug 11, 2013 |
| SCCC 1993 | Dec 15, 2017 |
| SCCC 1994 | Dec 15, 2017 |
| Shawn Chidester | Aug 9, 2016 |
| Shy | Jul 19, 2016 |
| SSS* and Dual* | Jan 22, 2018 |
| Stan Arts | Nov 21, 2014 |
| Stephen F. Wheeler | Jun 13, 2016 |
| Steven Edwards | Aug 26, 2017 |
| Steven Walczak | May 23, 2016 |
| Storm | Dec 9, 2016 |
| Thomas Petzke | Mar 15, 2012 |
| Tony Marsland | Nov 27, 2017 |
| Turbo Chess | Feb 19, 2017 |
| Turbostar | Jan 8, 2016 |
| Victor Vikhrev | Jan 7, 2016 |
| WCCC 1983 | Jan 20, 2018 |
| WCCC 1986 | Jul 27, 2017 |
| WCCC 1989 | Dec 5, 2017 |
| WCCC 1992 | Feb 26, 2018 |
| White Knight | Dec 12, 2016 |
| WMCCC 1980 | Dec 25, 2017 |
| WMCCC 1991 | Sep 18, 2016 |
| x86 | Jan 4, 2018 |
| Ziggy | Jan 12, 2016 |
| Zzzzzz | May 14, 2016 |

Up one Level