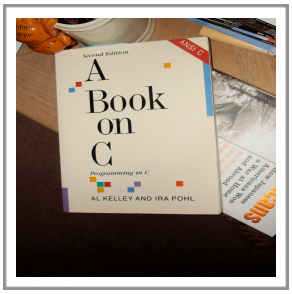
C Programming
Kyle Okamoto
Palm Desert Charter Middle Botball Team 11-0033

C Programming

int main()
{

C is a versatile language. You can use it to program computers, robots, and just about anything else electronic that has a "brain." It is easy to learn, but quite hard to master. I just started programming this year and my first program made the create go forward for 2 seconds. By this time I have learned most of the create library and some of the servo and motor programming as well. I have also started to dabble in computer programming, which is much harder. I read a book on programming which you can see in Picture #1. Later in my paper, you will see my own program in the My Programs section, but first, let me tell you where C Programming came from.



Picture 1 – A Book on C [16]

The History of C

C programming came around in about 1973. C was designed for implementing software; however, it is also used widely for portable application software, such as robots! C is one of the most popular programming languages of all time and some high schools are accepting it as a language like Spanish and French. It is quite hard to find a computer that doesn't have a "C compiler." C influenced many other languages, including KISS-C, C++, C--, Python, and much more.


Picture 2 -
An old computer for C [17]

C programming started off quite simply. It was a lot like KISS-C but instead of int main it was just main. You didn't have to put #include<stdio.h> or return(0) in a printf statement program. I wish it was this simple now because it would be a lot simpler to learn! The reason the format of C language keeps changing is because of the American National Standards Institute, ANSI for short. ANSI formed a committee called X3J11 in 1983 to establish a standard specification of C. In 1989, the standard was ratified as ANSI X3.158-1989 "Programming Language C." This version of the language is also referred to as ANSI C, Standard C, or C89. Then, in 1990, the ISO (International Organization for Standardization) adopted the ANSI standard and this version of C is referred to as C90, but C90 and C89 are exactly the same language. In picture 2, you can see an old

computer for C programming. Then in 1999, a new standard, C99, was introduced. This is the standard we write today.

Examples

An example of the early programming is this:

```
main()
{
        printf("hello, world!/n");
}
```

Another example of C with the new ANSI standard, C99:

```
#include<stdio.h>
int main()
{
        printf("hello, world!");
        return(0);
}
```

Recently a new standard, C1X, has been thought of and is being worked on.

Uses of C

As is stated before, C is a versatile language. Like the board game Othello, also known as Reversi, it is easy to learn but hard to master. C is often used for "system programming", including implementing operating systems and embedded system applications, due to a combination of desirable characteristics such as code efficiency, ability to access specific hardware addresses, and low runtime demand on system resources. C can also be used for website programming using CGI as a "gateway" for information between the Web application, the server, and the browser. Some reasons for choosing C over interpreted languages are its speed, stability, and near-universal availability. One consequence of C's wide acceptance and efficiency is that compilers, libraries, and interpreters of *other* programming languages are often implemented in C.

C is sometimes used as an intermediate language by implementations of other languages. This approach may be used for portability or convenience; by using C as an intermediate language, it is not necessary to develop machine-specific code generators. Some languages and compilers which have used C this way are BitC, C++, COBOL, Eiffel, Gambit, GHC, Squeak, and Vala. However, C was designed as a programming language, not as a compiler target language, and is thus less than ideal for use as an intermediate language. This has led to development of C-based intermediate languages such as C--.

C has also been widely used to implement end-user applications, but much of that development has shifted to newer languages.

Syntax

The syntax of C is a set of rules that specifies whether the sequence of characters in a file is conforming C source code. In other words, you could say that C has a formal grammar. C code is free-form, which means we can use whitespace to write comments. The C compiler always treats comments as whitespace.

Keywords

C89 has 32 keywords (reserved words with special meaning): `auto`, `break`, `case`, `char`, `const`, `continue`, `default`, `do`, `double`, `else`, `enum`, `extern`, `float`, `for`, `goto`, `if`, `int`, `long`, `register`, `return`, `short`, `signed`, `sizeof`, `static`, `struct`, `switch`, `typedef`, `union`, `unsigned`, `void`, `volatile`, and `while`.

C90 adds five more keywords: `inline`, `restrict`, `_Bool`, `_Complex`, and `_Imaginary`.

C supports a rich set of operators, which are symbols used within an expression to specify the manipulations to be performed while evaluating that expression. C has operators for:

- arithmetic: `+`, `-`, `*`, `/`, `%`
- assignment: `=`
    - augmented assignment: `+=`, `-=`, `*=`, `/=`, `%=`, `&=`, `|=`, `^=`, `<<=`, `>>=`
- bitwise logic: `~`, `&`, `|`, `^`
- bitwise shifts: `<<`, `>>`
- boolean logic: `!`, `&&`, `||`
- conditional evaluation: `? :`
- equality testing: `==`, `!=`
- calling functions: `( )`
- increment and decrement: `++` and `--`
- member selection: `.`, `->`
- object size: `sizeof`
- order relations: `<`, `<=`, `>`, `>=`
- reference and dereference: `&`, `*`, `[ ]`
- sequencing: `,`
- subexpression grouping: `( )`

- type conversion: `(typename)`

Languages Influenced by C

Some programming languages influenced by C are Python, C++, C--, Go, Perl, C#, Objective C, Java, JavaScript, PHP, LPC, and Unix's C Shell. The most pervasive influence has been syntactical: all of the languages mentioned combine the statement and (more or less recognizably) expression syntax of C with type systems, data models and/or large-scale program structures that differ from those of C, sometimes radically. Go was developed by the mega-company Goole. Personally, I like Go the best because it eliminates parentheses and semicolons, which I tend to forget a lot.

My Programs

Picture 3- A CBC Botball Controller[15]

I wrote a program for the create, and I am currently working on a program for the computer. The CBC, shown in Picture 3, is the brain where you download the programs to control Botball robots. I am a sixth grader and have learned KISS-C this year. I have found KISS-C is a very easy language to learn, and these are my programs.

Create Program:

This program makes the create go forward, fetch the balls, and take them to the holding area for biofuels. This is not reliable, though. Once I crashed into the runway.

```
int main()
  {
    create_connect();
    create_full();
    create_drive_straight(300);
    sleep(2);
    set_servo_position(1,2040);
    set_servo_position(2,240);
    create_spin_CW(200);
    sleep(1.03);
    create_drive_straight(100);
    sleep(5);
    create_spin_CW(300);
    sleep(1);
    create_drive_straight(-100);
    sleep(2);
```

```
    create_spin_CCW(400);
    sleep(1);
    create_drive_straight(-50);
    sleep(5);
    create_spin_CCW(300);
    sleep(1);
    create_drive_straight(200);
    sleep(3);
    create_spin_CW(300);
    sleep(2);
    create_drive_straight(500);
    sleep(2);
  }
```

Picture 4 -
This is me programming! [18]

Computer Program:

This program makes the computer say “hello, everybody!”

In picture 4 you can see me programming on my laptop.

```
#include <stdio.h>
int main()
  {
  printf("hello, everybody!”)
  return(0);
  }
```

Summary

This is what I know about C. I like this programming language a lot but I would like to see the parentheses disappear. I personally don’t like the semicolons too, but those are okay. I hope to one day master C and see “Go” mature into a great language. I like Go a lot because it really simplifies C. I also hope to create a similar language to C.

}

Bibliography

1. ^ *a b* Kernighan; Dennis M. Ritchie (March 1988). The C Programming Language (2nd ed.).Englewood Cliffs, NJ: Prentice Hall. ISBN 0-13-110362-8. http://cm.bell-labs.com/cm/cs/cbook/. Regarded by many to be the authoritative reference on C.
2. ^ *a b* Giannini, Mario; Code Fighter, Inc.; Columbia University (2004). "C/C++". In Hossein Bidgoli. *The Internet encyclopedia*. **1**. John Wiley and Sons. p. 164. ISBN 0471222011.
3. ^ *a b* Dennis M. Ritchie (January 1993). "The Development of the C Language". http://cm.bell-labs.com/cm/cs/who/dmr/chist.html. Retrieved Jan 1 2008. "The scheme of type composition adopted by C owes considerable debt to Algol 68, although it did not, perhaps, emerge in a form that Algol's adherents would approve of."
4. ^ van Rossum, Guido (1993). "An Introduction to Python for UNIX/C Programmers". *Proceedings of the NLUUG najaarsconferentie (Dutch UNIX users group)*. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.38.2023. "even though the design of C is far from ideal, its influence on Python is considerable.".
5. ^ Patricia K. Lawlis, c.j. kemp systems, inc. (1997). "Guidelines for Choosing a Computer Language: Support for the Visionary Organization". Ada Information Clearinghouse. http://archive.adaic.com/docs/reports/lawlis/k.htm. Retrieved 2006-07-18.
6. ^ "Programming Language Popularity". 2009. http://www.langpop.com/. Retrieved 2009-01-16.
7. ^ "TIOBE Programming Community Index". 2009. http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html. Retrieved 2009-05-06.
8. ^ *a b* Kernighan, Brian W.; Dennis M. Ritchie (February 1978). *[[The C Programming Language (book)|]]* (1st ed.). Englewood Cliffs, NJ: Prentice Hall. ISBN 0-13-110163-3. The first widely available book on the C programming language.
9. ^ *Dr. Dobb's Sourcebook*. U.S.A.: Miller Freeman, Inc.. Nov/Dec 1995 issue.
10. ^ "Using C for CGI Programming". linuxjournal.com. 2005-03-01. http://www.linuxjournal.com/article/6863. Retrieved 2010-01-04.
11. ^ Harbison, Samuel P.; Guy L. Steele (2002). *C: A Reference Manual* (5th ed.). Englewood Cliffs, NJ: Prentice Hall. ISBN 0-13-089592-X. Contains a BNF grammar for C.
12. ^ "10 Common Programming Mistakes in C". Cs.ucr.edu. http://www.cs.ucr.edu/~nxiao/cs10/errors.htm. Retrieved 2009-06-26.
13. ^ Page 6 of the original K&R[8]
14. ^ ISO/IEC 9899:1999 specification, p. 47, § 6.3.2.3 (3)
15. www.botball.org/cbc
16. Original photo taken by Kyle Okamoto
17. blogs.sfweekly.com
18. Original photo taken by Kyle Okamoto