PFORD Language: An Overview

Bhagyashree Bohra
Deepti Jindal (group leader)
Sharib Khan
Shringika Porwal

Introduction

PFORD (programming for dummies) is a language targeting both novice and advanced programmers.

The motivation behind PFORD is to provide an easy to learn, easy to understand language which can be used by novices to quickly familiarize themselves with the art of programming and logical thinking. High level languages like C and Java often present a formidable challenge to novices who get bogged down by the non-intuitive presentation of the languages. One has to spend substantial time just understanding the syntax before being able to write a single line of code. PFORD has been designed to overcome this learning curve by using an easy to understand and self-explanatory syntax and semantics. The language employs simple representative English words in the syntax and permits specification of statements that are akin to describing the algorithm to solve the problem at hand. This simplicity will enable a novice to solve problems using high level concepts and not worry too much about the programming details.

PFORD also has another neat feature. The compiler can be instructed to generate equivalent code in any of the popular languages like Java and C. A user can easily learn PFORD and simply translate to C or Java without having to spend time learning the corresponding languages.

Language

PFORD is a simple, intuitive, easy to learn, user-friendly, and a powerful language, which also has the capability to translate into other high level popular languages.

Ease-of-Use

The goal of PFORD was to build a system that would be simple for the advanced as well as the novice user. Each language has its own advantages and one may be more suitable than the rest for a project. Programming with PFORD, the user needs to write a program only once and then can generate the corresponding Java or C code by setting a flag at the beginning of the program. It will be possible to test the result over each platform to observe which of them provides the most optimal result. We also wanted to simplify the process of learning programming for beginners. By making syntax of the commands similar to English language we have eliminated many of the confusing features of Java/ C and have made it easier for the user with minimal programming experience to comprehend PFORD.

For example a loop in C/Java
for(int i=0; i<MAX_VALUE; i++)...

PFORD is syntactically much simpler and visually much easier to understand.

loop till MAX_VALUE

Object-Oriented

PFORD will allow the user the capacity of creating user-defined classes and let the user generate instances of their objects. This feature will provide the capability of building customized data types which can be reused, making programming easier.

Powerful

PFORD offers a beginner the power to perform complex tasks by writing fewer lines of code. This language provides compact commands to perform complex mathematical and string manipulation operations, and also allows the creation of user defined objects. Using PFORD, the programmer has the ability to translate code to Java or C making it a powerful tool.

Efficient

PFORD was created to save time and resources for experienced programmers and to eliminate the burden of learning complex languages for the novice users. A single language can now be used to generate code in any of the commonly used high level languages.

Customizable

Not only does PFORD provide the ability to translate C/Java, but the user can also run their program with this compiler. It can be used to create customized objects, classes and functions.

Robust

PFORD has been developed as a dependable language. It allows compile-time checks to detect the errors that would be found by C and Java before generating the corresponding code, thus preventing errors while running the programs on their respective platforms. Another feature of PFORD is that memory management is inherently handled by Java thereby eliminating problems related to memory leaks and making PFORD more robust.

Architecture neutral

PFORD code is compiled to Java code, which is interpreted by Java Virtual Machine (JVM). JVM is a machine independent platform, this makes Java portable to virtually any machine. As PFORD is based on Java, this provides PFORD, the quality of being architecture neutral. PFORD Compiled code can be executed on any processor/platform wherever JVM will run.

Data Types

PFORD supports several data types such as *integer, float, char and string*. These data types follow the same fundamental pattern as C/Java. In addition, user defined objects, classes, arrays and hashes are also provided. Arrays are used to represent any of the data types listed above.

Basic Functions/ Libraries

The three basic libraries in PFORD are math, sorting and I/O file. Some of the functions supported include cosine, sine, log, pi, square root, floor, ceiling and sorts on integer and string arrays.

Interpreted

The lexer and parser will be created using ANTLR. ANTLR will generate the required information for PFORD interpreter to execute the user's program. Since PFORD language is not compiled to machine code, it's interpreter will work the same on all the platforms and architectures.

Sample Code

A sample program in PFORD that does translation would look like:

startofprogram
Generate C

Variable a *type int*

```
a=0

Loop 10 runs
Increment a

Output "Final value" a
endofprogram;
```

When this program will be run it will output the corresponding C code.

```
void main()
{
int a;
a=0;

for (int i; i<11; i++)
  a++;

cout << "Final value " << a << endl;
}
```

An example of a program that will run in PFORD would look like:

```
/*indicates the beginning of the program */
startofprogram;

Include library math.lib;

Integer a ,n = 0;
Decimal b = 0;
Boolean c = 0;
/* defines an array of integers */
Array Integer  d_array= [1,2,3,5,6];

/* Looping and adding a to itself 10 times*/
Loop 10 runs {a++};
Output a;

/*Conditional statements.*/
if valid (b = = 0)
        then do {b = b+1.0;}
else do {b = b.0 –1.0;}

/*Conditional loops*/
while valid (c= =0) loop through{
        c = 1;
}

loop through {
        c =  0;
} while valid ( c = = 1);

/*Looping till we reach a stopping value*/
loop till (a = 20) {a++};


/* Iterative "for" loop */
loop from (n==0) till (n>10) changing(n = n+1) {
        Output n;
}

/ *calls the sort function of the math library with the array as an argument* /.
call function sort with args{d_array};

/*indicates the end of the program */
endofprogram;
```

The italics indicate keywords in the language. Since the reference manual is still not complete, the final syntax used may differ slightly.

Future

In the words of Alfred North Whitehead, "Civilization advances by extending the number of important operations which we can perform without thinking." The whole philosophy behind PFORD adds a little to this maxim. We believe that one should be able to do the maximum with the minimum amount of coding.
To facilitate this, future versions of PFORD will support all languages instead of just C and Java. Ideally we would like to be supporting Perl, Python, Awk, etc….

The power of PFORD. The following snippet prints out common values between two files and also values found in one and not the other- all in just 6 lines.

```
        PERL:
        Set A= read_file ("file.txt", "|", 0, "\n");
        Set B= read_file ("file1.txt", "|", 0,"\n");
        Set common= A and B;
        Set diff_A= A-common;
        PFORD(future version):
        Output "Elements common to both file and file1 are \n", common;
        Output "Elements in file that are not in file1 are \n", diff_A;
```

Another future development is PFORD will provide some highly easy to use and optimized libraries for creating complex data structures and performing string manipulations. These features are not supported by C and Java's version is not very intuitive. Hence, we have incorporated simple to use data structures, namely arrays, hashes and sets which have been inspired by the PERL programming language. Some elements include stacks, queues and binary tree functions.

Summary

PFORD is quick, simple and easy to learn. It provides a powerful edge over the other programming languages because it enables the novice to learn programming effortlessly as well as provide advanced programmers the power to write code once and switch to C/Java easily. It gives the users 3 languages at their disposal, and in future it will give them all languages and abstract complicated code as single word functions.
