Proposal for EasyDraw programming language

Stanislav Ivaciov

COMS W4115 – Spring 2009 (CVN)
Dr. Edwards

Introduction

EasyDraw is a simple programming language. It describes a process of drawing two dimensional geometric figures on the screen.

The language is designed to facilitate learning of general programming concepts by novices. EasyDraw accomplishes this objective by associating programming constructs and concepts with graphical representations.

Similar to BASIC, EasyDraw provides advanced but easy to understand concepts. But instead of text oriented output the language produces pictures. Not to underestimate it's power, EasyDraw can be used for drawing blueprints and graphs of mathematical functions.

Language Overview

EasyDraw programming language is implemented in an interpreter. To construct a program the authors create a single file that can contain definitions of multiple macros.

Program execution always begins from the first line of the macro called "main" from which other macros can be called (that includes recursive calls).

Some of the constructs include: repetition (while loops), variables of types (integers and colors), macro calls, decision making (if … then … else …), calls to library macros (e.g.: draw is the only one available as of now), higher level primitives (line, circle, square, rectangle, point, polygon, triangle) that are passed to "draw" as a parameter.

Macros do not take arguments. Argument passing is simulated through global variables manipulations.

Code for macros and constructs is defined starting from '{' and ending '}' only global variables can be defined outside of macros.

Example Program

Color conventions:

Macro definitions and calls
Reserved words
Comments

File: SampleProg.ed

```
/* definition of global variables, accessible in any macro */
integer  i      = 4;    /* global variable of type integer */
color    c      = 0;    /* variable of type color (0 - black) */
integer  radius = 100;


/* definition of macro "draw_circles" */
draw_circles {
   integer i = 5;    /* local variable */
   while (i >= 0) {
      call draw circle 50  50 radius;
      radius = radius - 1;
      i = i - 1;
   }
}


/* definition of macro "main", every program
   begins its execution in macro called main */
main {
   if (i > 0) {
      i = i - 1;          /* decrement global variable i */
      c = c + 1;
      call draw_circles;  /* call draw_circles macro */
      call main;          /* recursive call */
   }
   else {
      c = 0;
      call draw square 50 50 45;
   }
}

EOF
```

Output: