GD::Graph - Graph Plotting Module

José Pedro Silva Pedro Faria Ulisses Costa

Engenharia de Linguagens
Projecto integrado

January 10, 2011

Description

GD::Graph is a perl5 module to create charts using the GD module. The following classes for graphs with axes are defined:

GD::Graph::lines Create a line chart.

GD::Graph::bars Create a bar chart with vertical or horizontal bars.

GD::Graph::linespoints Combination of lines and points.

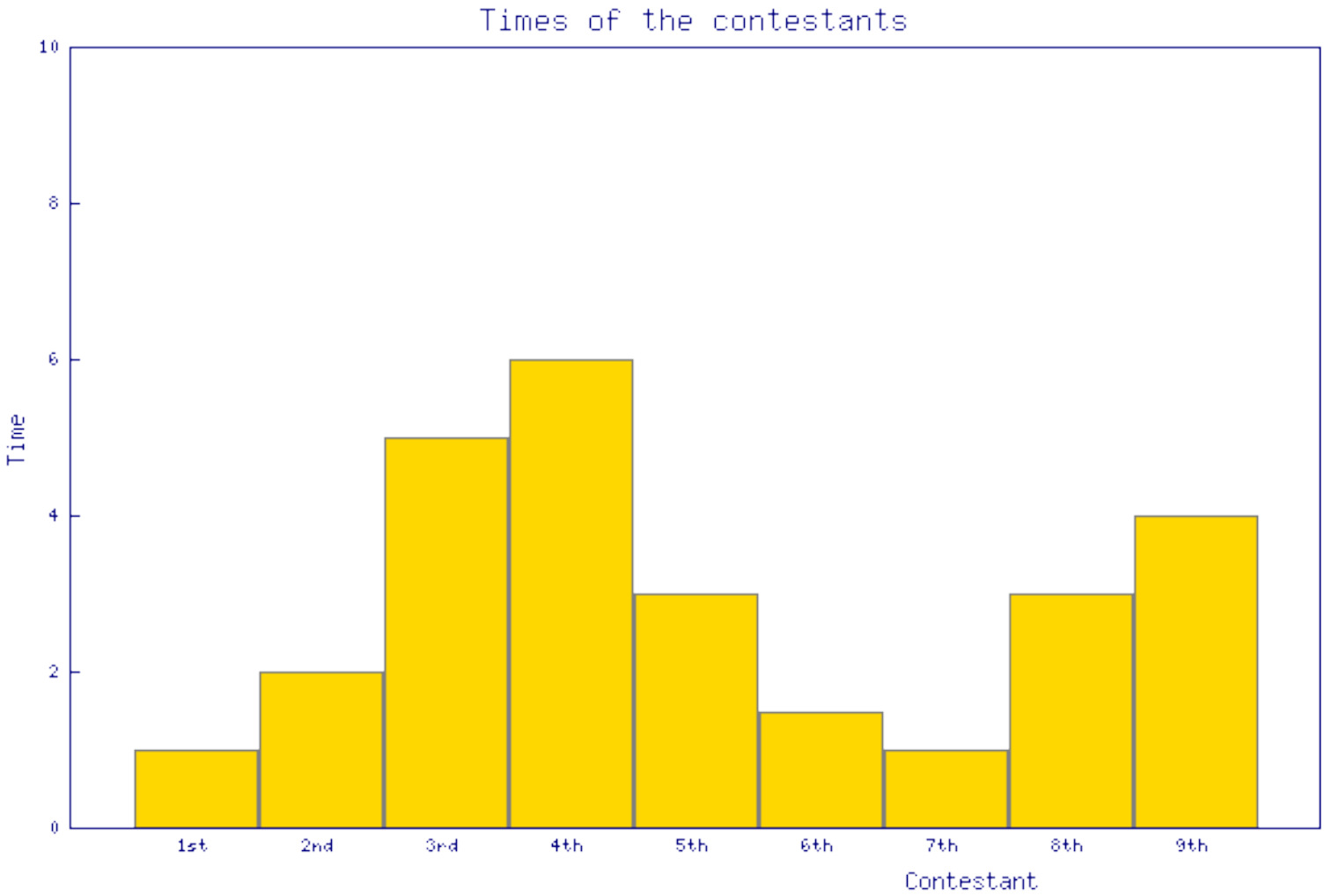
GD::Graph::area Create a graph, representing the data as areas under a line.

GD::Graph::pie Create a pie chart.

Bars

```
1   sub plotPng {
2       my $fileName = shift;
3       @data = ( ["1st","2nd","3rd","4th","5th","6th","7th", "8th", "9th"],
4                 [    1,    2,    5,    6,    3,  1.5,    1,     3,     4]
5               );
6       my $mygraph = GD::Graph::bars->new(600, 400);
7
8       $mygraph->set (
9           x_label         => "Contestant",
10          y_label         => "Time",
11          title           => "Times of the contestants",
12          dclrs           =>  [ qw(gold red green) ],
13      ) or warn $mygraph->error;
14
15      my $myimage = $mygraph->plot(\@data) or warn $mygraph->error;
16
17      open (IMG, '>' , $fileName);
18      binmode IMG;
19      print IMG $myimage->png;
20      close (IMG);
21  }
```

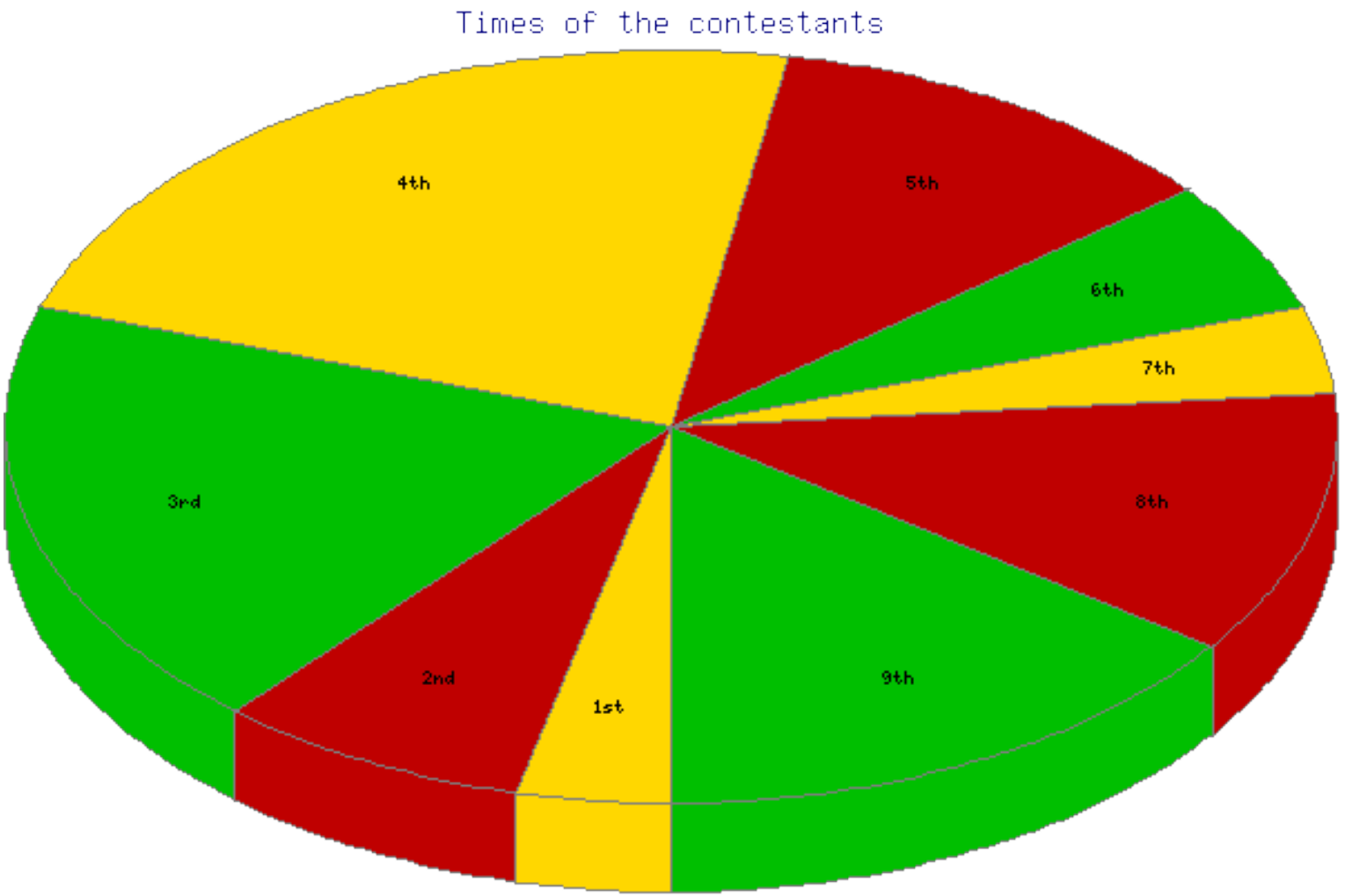
Bars - Output


Times of the contestants
Time
0
2
4
6
8
10
1st
2nd
3rd
4th
5th
6th
7th
8th
9th
Contestant

Pie

```
1   sub plotPng {
2       my $fileName = shift;
3       @data = ( ["1st","2nd","3rd","4th","5th","6th","7th", "8th", "9th"],
4                 [    1,    2,    5,    6,    3,  1.5,    1,     3,     4]
5               );
6       my $mygraph = GD::Graph::pie->new(600, 400);
7
8       $mygraph->set (
9           title          => "Times of the contestants",
10          dclrs          =>  [ qw(gold red green) ],
11      ) or warn $mygraph->error;
12
13      my $myimage = $mygraph->plot(\@data) or warn $mygraph->error;
14
15      open (IMG, '>' , $fileName);
16      binmode IMG;
17      print IMG $myimage->png;
18      close (IMG);
19  }
```

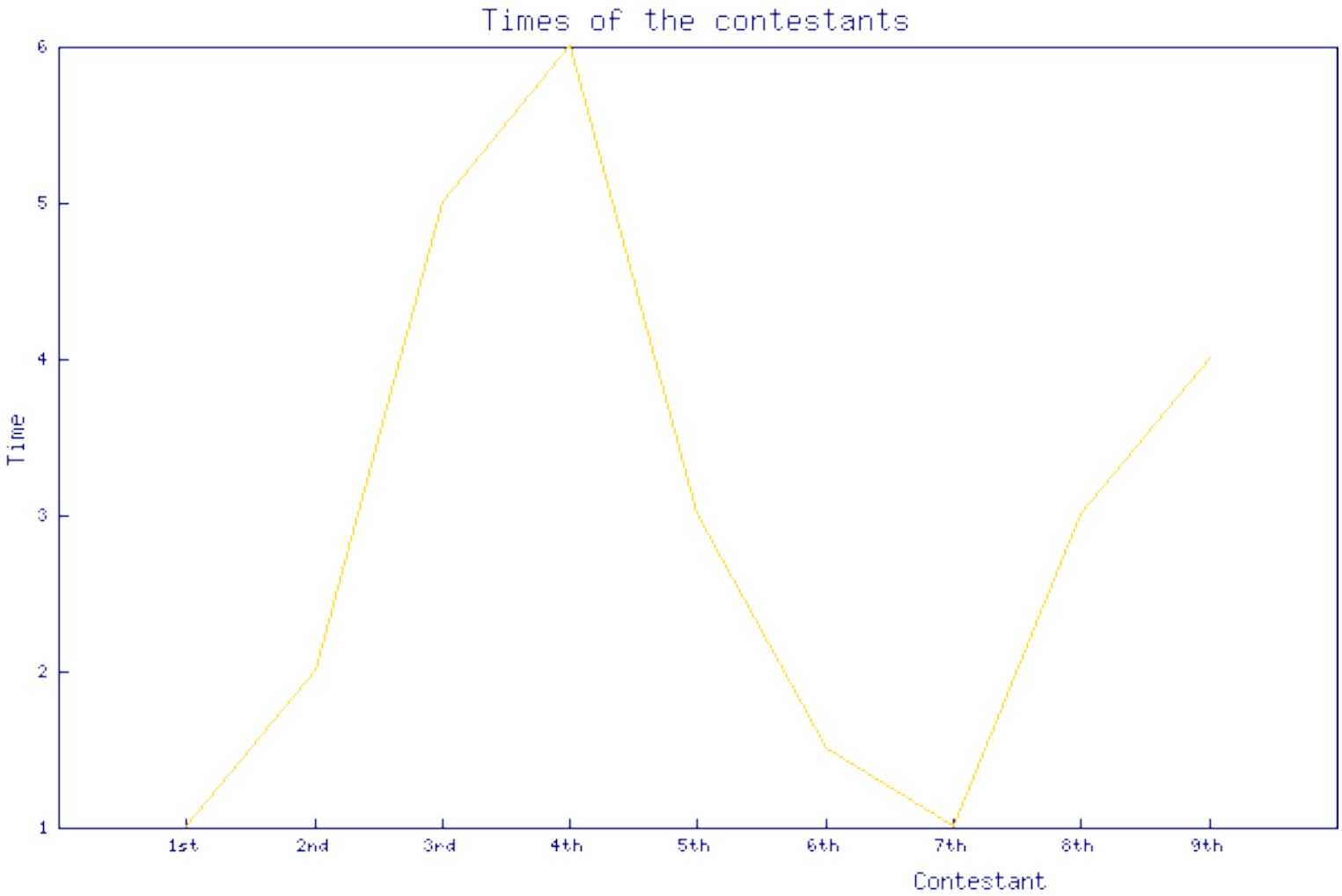
Pie - Output



Line

```
1   sub plotPng {
2       my $fileName = shift;
3       @data = ( ["1st","2nd","3rd","4th","5th","6th","7th", "8th", "9th"],
4                 [    1,    2,    5,    6,    3,  1.5,    1,     3,     4]
5               );
6       my $mygraph = GD::Graph::area->new(600, 400);
7
8       $mygraph->set (
9           title          => "Times of the contestants",
10          dclrs          =>  [ qw(gold red green) ],
11      ) or warn $mygraph->error;
12
13      my $myimage = $mygraph->plot(\@data) or warn $mygraph->error;
14
15      open (IMG, '>' , $fileName);
16      binmode IMG;
17      print IMG $myimage->png;
18      close (IMG);
19  }
```

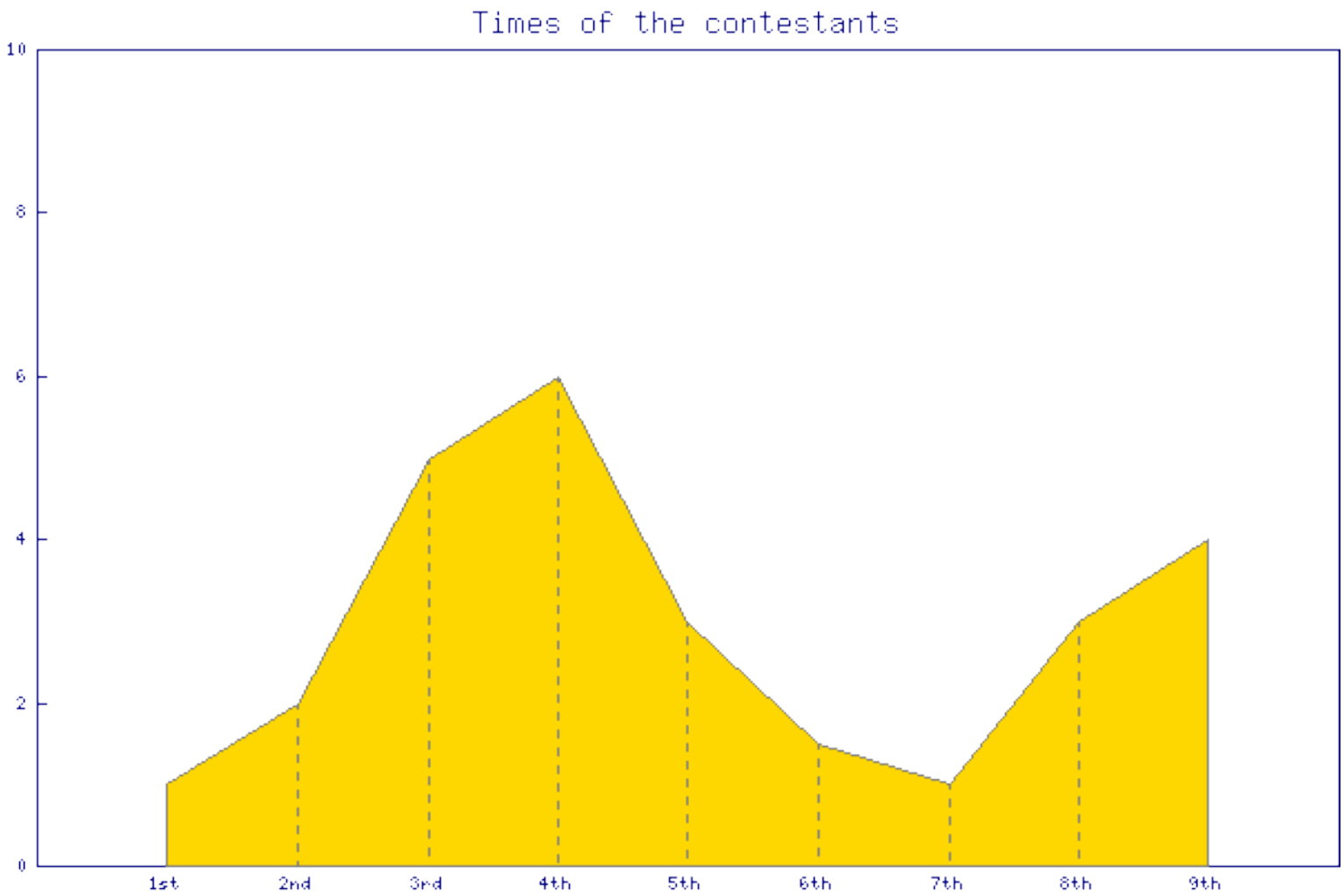
Line - Output


Times of the contestants
Time
6
5
4
3
2
1
1st
2nd
3rd
4th
5th
6th
7th
8th
9th
Contestant

Area

```
1   sub plotPng {
2       my $fileName = shift;
3       @data = ( ["1st","2nd","3rd","4th","5th","6th","7th", "8th", "9th"],
4                 [    1,    2,    5,    6,    3,  1.5,    1,     3,     4]
5               );
6       my $mygraph = GD::Graph::area->new(600, 400);
7
8       $mygraph->set (
9           title          => "Times of the contestants",
10          dclrs          =>  [ qw(gold red green) ],
11      ) or warn $mygraph->error;
12
13      my $myimage = $mygraph->plot(\@data) or warn $mygraph->error;
14
15      open (IMG, '>' , $fileName);
16      binmode IMG;
17      print IMG $myimage->png;
18      close (IMG);
19  }
```

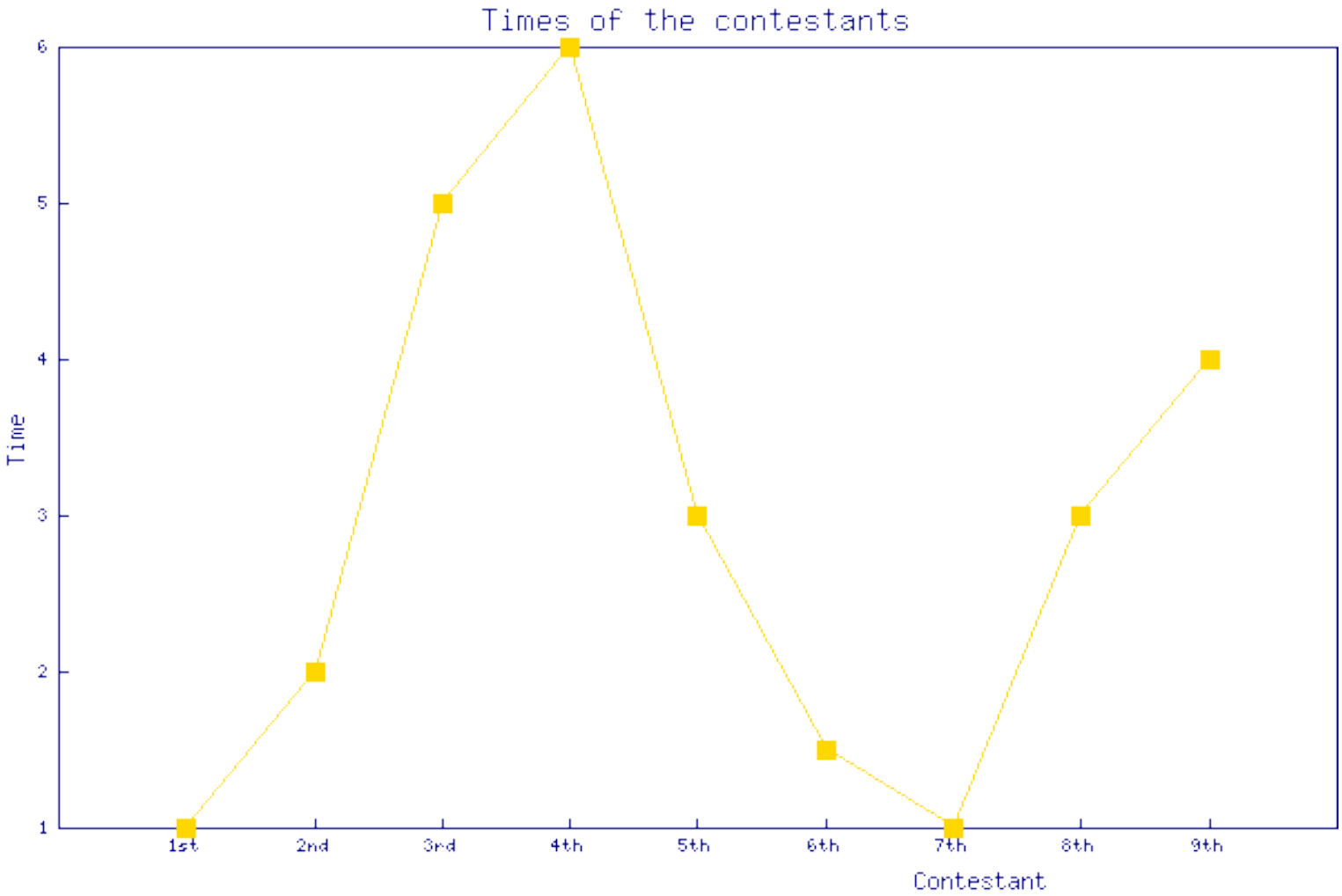
Area - Output


Times of the contestants
10
8
6
4
2
0
1st
2nd
3rd
4th
5th
6th
7th
8th
9th

Points and Lines

```
1   sub plotPng {
2       my $fileName = shift;
3       @data = ( ["1st","2nd","3rd","4th","5th","6th","7th", "8th", "9th"],
4                 [    1,    2,    5,    6,    3,  1.5,    1,     3,     4]
5               );
6       my $mygraph = GD::Graph::points->new(600, 400);
7
8       $mygraph->set (
9           title          => "Times of the contestants",
10          dclrs          =>  [ qw(gold red green) ],
11      ) or warn $mygraph->error;
12
13      my $myimage = $mygraph->linespoints(\@data) or warn $mygraph->error;
14
15      open (IMG, '>' , $fileName);
16      binmode IMG;
17      print IMG $myimage->png;
18      close (IMG);
19  }
```

Points and Lines - Output



Examples of output formats

```
1   print IMG $graph->plot(\@data)->gif;
2   print IMG $graph->plot(\@data)->png;
3   print IMG $graph->plot(\@data)->gd;
4   print IMG $graph->plot(\@data)->gd2;
```

Methods for all graphs

GD :: Graph :: chart− > new(width, height) Create a new object *graph* with optional width and heigth. Default *width* = 400, default *height* = 300. chart is either bars, lines, points, linespoints, area, mixed or pie.

graph− > set_text_clr(colourname) Set the colour of the text. This will set the colour of the titles, labels, and axis labels to colour name. Also see the options *textclr*, *labelclr* and *axislabelclr*.

graph− > set_title_font(fontspecification) Set the font that will be used for the title of the chart.

graph− > plot(data) Plot the chart, and return the GD::Image object.

Methods for all graphs - 2

graph− > *set*(*attrib*1 => *value*1, *attrib*2 => *value*2...) Set chart options.

graph− > *get*(*attrib*1, *attrib*2) Returns a list of the values of the attributes. In scalar context returns the value of the first attribute only.

graph− > *gd*() Get the GD::Image object that is going to be used to draw on. You can do this either before or after calling the plot method, to do your own drawing.

Methods for all graphs - 3

graph− > export_format() Query the export format of the GD library in use. In scalar context, it returns 'gif', 'png' or undefined, which is sufficient for most people's use. In a list context, it returns a list of all the formats that are supported by the current version of GD. It can be called as a class or object method

graph− > can_do_ttf() Returns true if the current GD library supports TrueType fonts, False otherwise. Can also be called as a class method or static method.

Options for graphs with axes

x_label, *y_label* The labels to be printed next to, or just below, the axes. Note that if you use the *two_axes* option that you need to use *y1_label* and *y2_label*.

show_values Set this to 1 to display the value of each data point above the point or bar itself.

values_space Space to insert between the data point and the value to print. Default: 4.

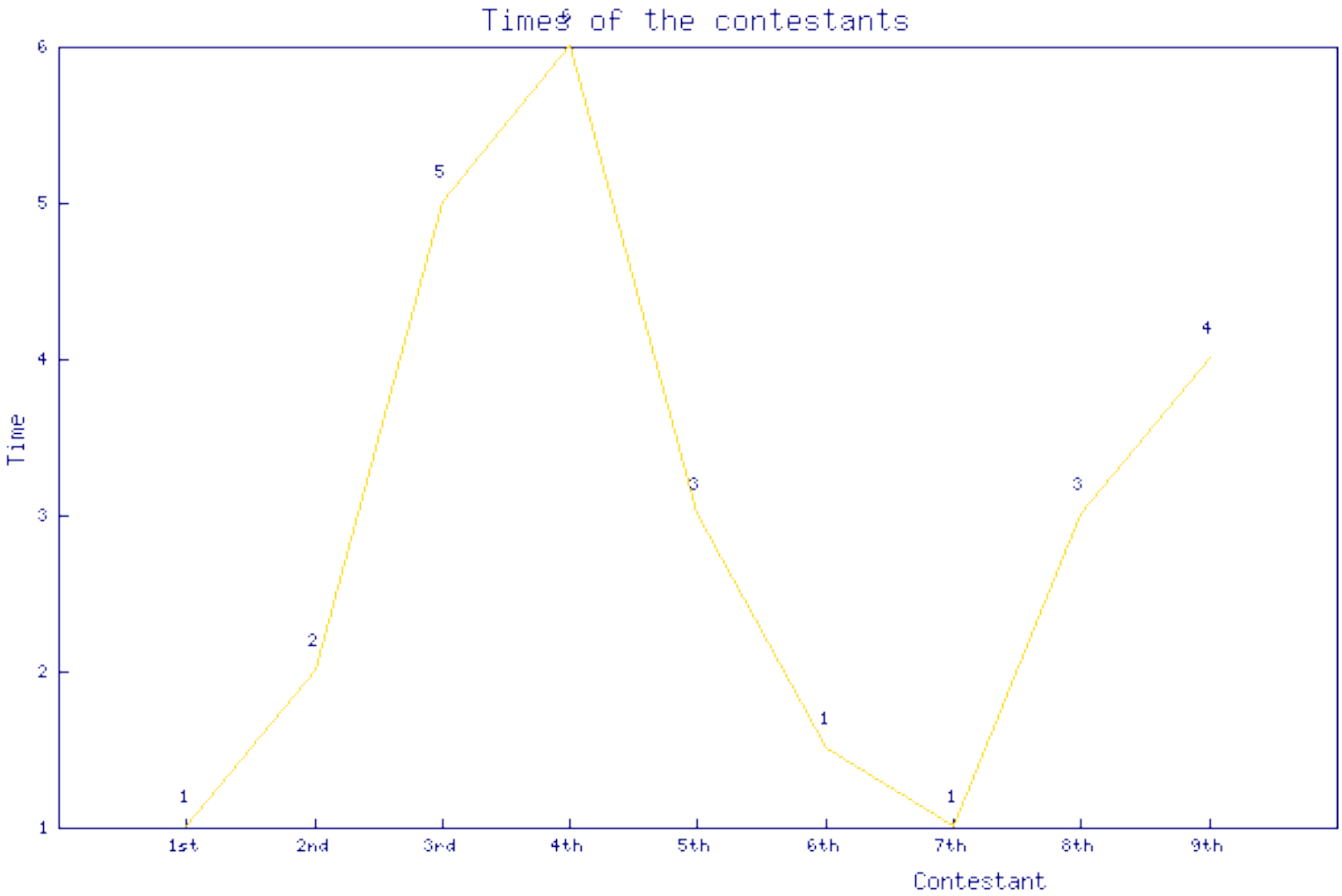
values_format How to format the values for display.

and many more...

Lines with *show_values*

```
1       $mygraph->set (
2           x_label => "Contestant",
3           y_label => "Time",
4           values_format => sub { return sprintf("\%d", shift); } ,
5           values_space  => 10,
6           show_values   => 1,
7           title          => "Times of the contestants",
8           dclrs          =>  [ qw(gold red green) ]
9       ) or warn $mygraph->error;
```

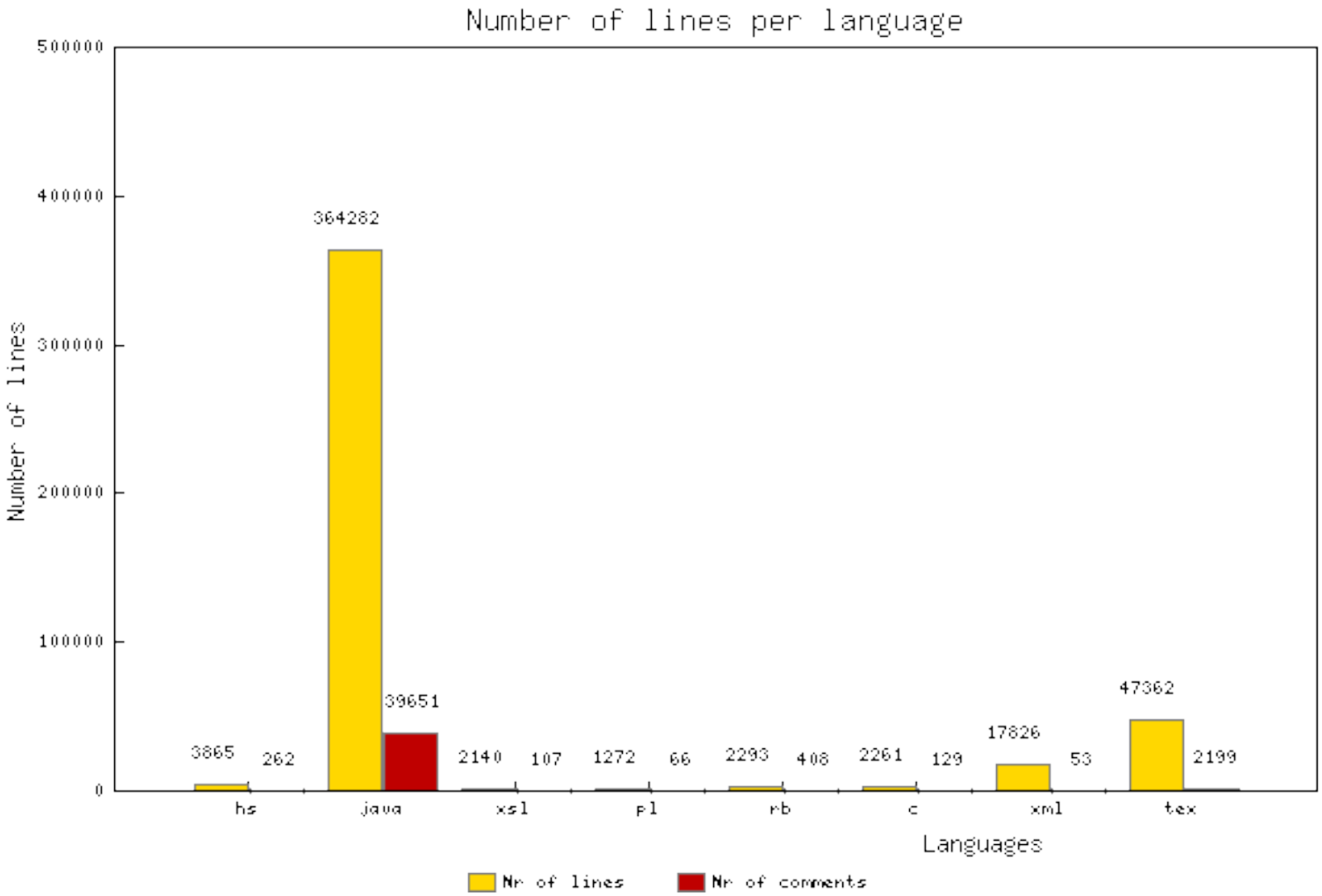
Lines with *show_values*



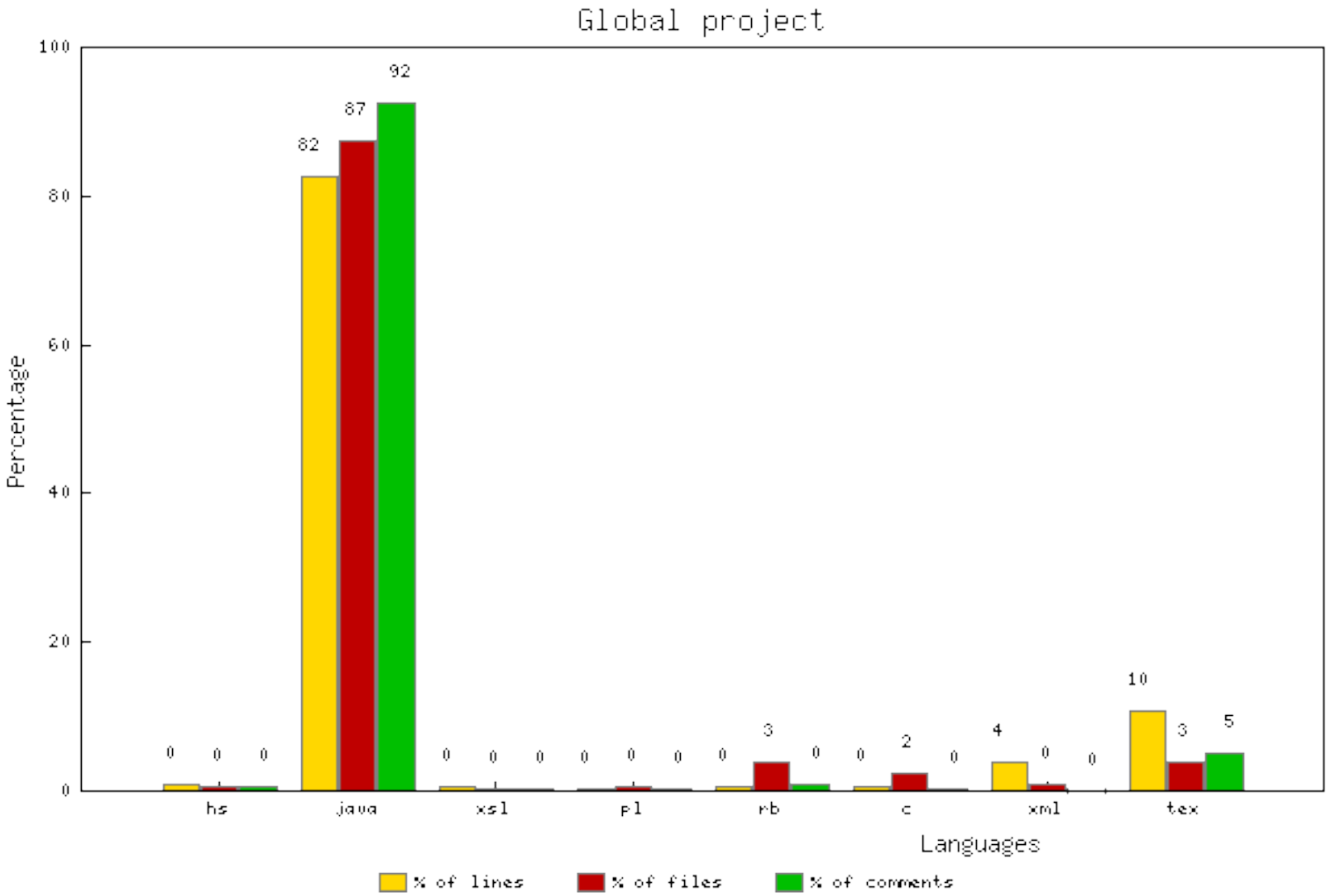
Legends and multiple Y values

```
1       my @legend_keys = ("Nr of lines","Nr of comments");
2       $mygraph->set_legend(@legend_keys);
3       $mygraph->set(
4           transparent   => 1,
5           overwrite => 0,
6           fgclr => black ,
7           labelclr => black,
8           axislabelclr => black,
9           legendclr => black,
10          valuesclr => black,
11          textclr => black,
12          transparent   => 1,
13          overwrite => 0,
14          bargroup_spacing => 10,
15          show_values   => 1,
16          values_format => sub { return sprintf("\%d", shift); } ,
17          values_space  => 10,
18          x_label       => $x_label,
19          y_label       => $y_label,
20          title         => $title,
21          dclrs         =>  [ qw(gold red green) ],
22      ) or warn $mygraph->error;
```

Legends and multiple Y values - Output



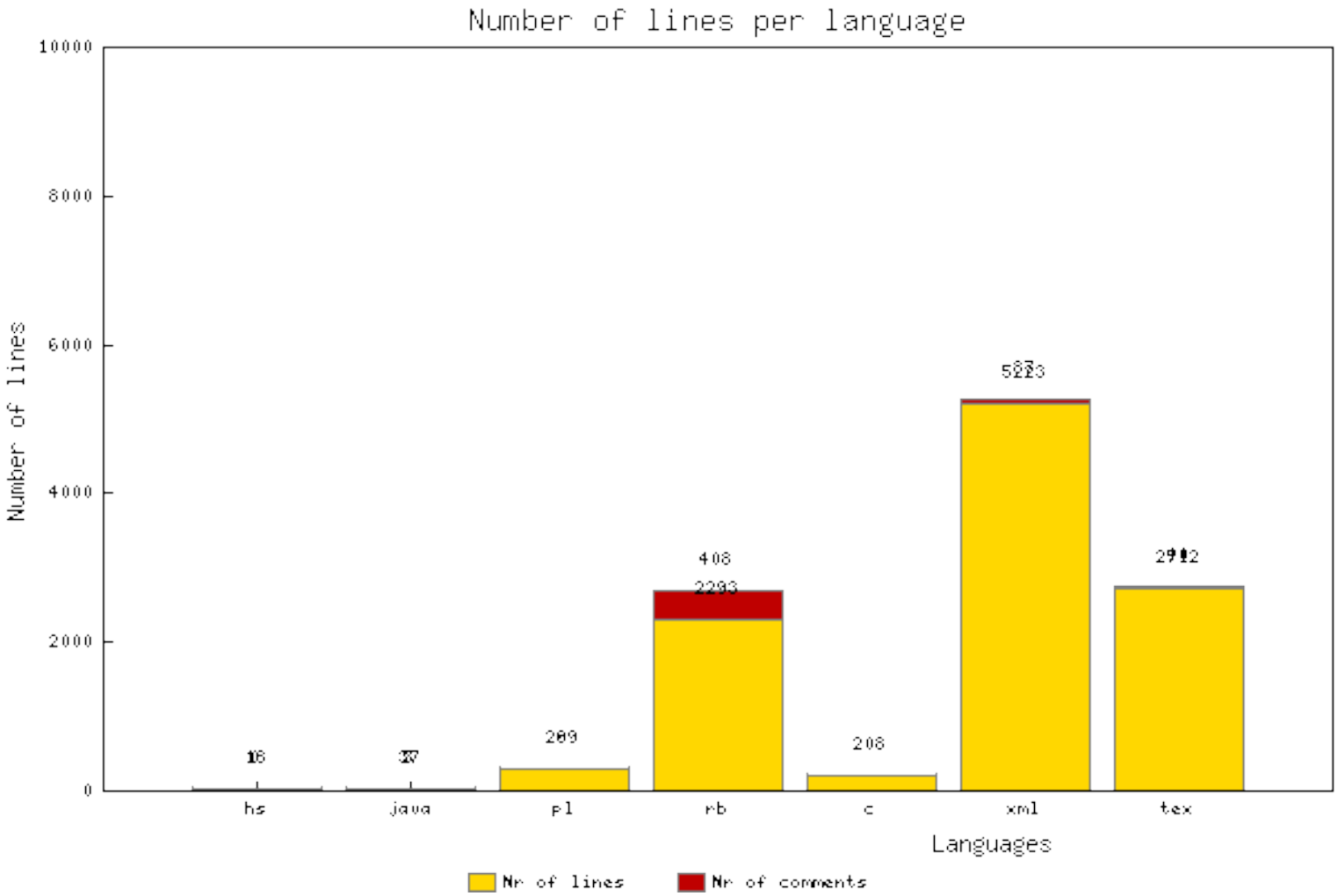
Legends and multiple Y values 2 - Output



Overwrite bars

```
1       my @legend_keys = ("Nr of lines","Nr of comments");
2       $mygraph->set_legend(@legend_keys);
3       $mygraph->set(
4           transparent   => 1,
5           overwrite => 2,
6           fgclr => black ,
7           labelclr => black,
8           axislabelclr => black,
9           legendclr => black,
10          valuesclr => black,
11          textclr => black,
12          transparent   => 1,
13          overwrite => 0,
14          bargroup_spacing => 10,
15          show_values   => 1,
16          values_format => sub { return sprintf("\%d", shift); } ,
17          values_space  => 10,
18          x_label       => $x_label,
19          y_label       => $y_label,
20          title         => $title,
21          dclrs         =>  [ qw(gold red green) ],
22      ) or warn $mygraph->error;
```

Overwrite bars - Output



Perguntas

?