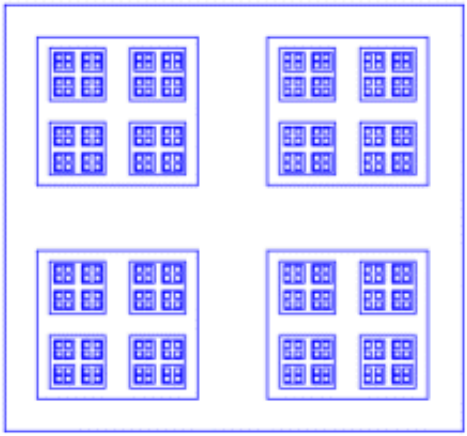
Zadání

Formulujte jednoduché rekurzivní algoritmy, pomocí kterých nakreslíte následující obrázky. Předpokládejte, že příkazy pro kreslení primitiv (úsečky, čtverce, apod.) jsou k dispozici.

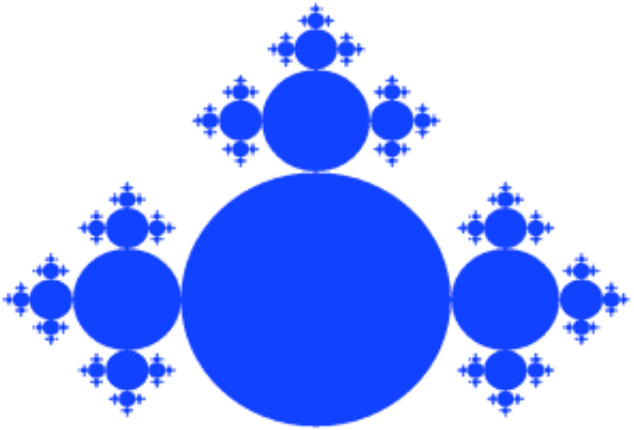
1. čtverce poprvé – předloha



1. čtverce poprvé – program

```
void draw_squares(int level, int x, int y, int a) {
   int new_a = a * .33;
   /* velky ctverec */
   printf("draw_rect %d %d %d %d\n", x, y, a, a);
   /* 4 mensi uvnitr */
   if (level++ <= 5) {
      draw_squares(level, x + a * 1./4. - new_a / 2,
            y + a * 1./4. - new_a / 2, new_a);
      draw_squares(level, x + a * 3./4. - new_a / 2,
            y + a * 1./4. - new_a / 2, new_a);
      draw_squares(level, x + a * 1./4. - new_a / 2,
            y + a * 3./4. - new_a / 2, new_a);
      draw_squares(level, x + a * 3./4. - new_a / 2,
            y + a * 3./4. - new_a / 2, new_a);
   }
}
```

2. Kruhy – předloha



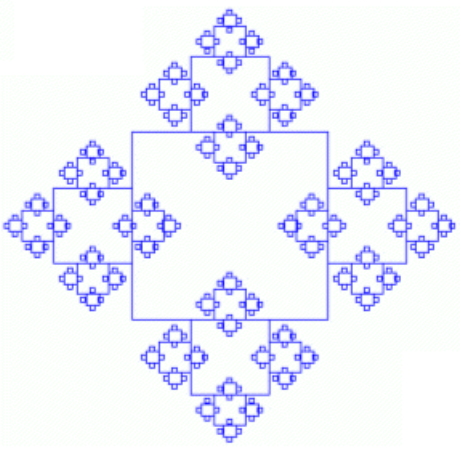
2. Kruhy – program 1./2

```
void draw_circles(int level, int x, int y,
      int r, int orient)
{
   int new_r = r / 2;
   printf(”draw_arc_f %d %d %d %d 0 360\n”,
         x, y, r, r);

   if (level++ <= 7) {
      if (orient != DOWN)
         draw_circles(level, x + r / 2 - new_r/2,
            y - new_r, new_r, UP);

      if (orient != UP)
         draw_circles(level, x + r / 2 - new_r/2,
            y + r, new_r, DOWN);
```

2. Kruhy – program 2./2

```
      if (orient != RIGHT)
         draw_circles(level, x - new_r,
            y + r / 2 - new_r / 2, new_r, LEFT);

      if (orient != LEFT)
         draw_circles(level, x + r,
            y + r / 2 - new_r / 2, new_r, RIGHT);
   }
}
```

3. Čtverce podruhé – předloha



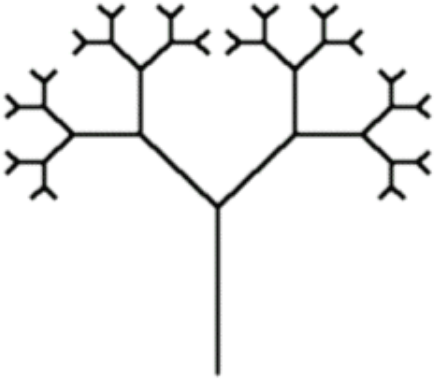
3. Čtverce podruhé – program 1./2

```
void draw_squares(int level, int x, int y, int r) {
   int new_r = r * 0.4;

   /* prostredni */
   printf(”draw_rect %d %d %d %d\n”, x, y, r, r);

   /* 3 okolo */
   if (level++ <= 5) {
      draw_squares(level, x + r / 2 - new_r/2,
            y - new_r, new_r);

      draw_squares(level, x + r / 2 - new_r/2,
            y + r, new_r);
```

3. Čtverce podruhé – program 2./2

```
        draw_squares(level, x - new_r,
                y + r / 2 - new_r / 2, new_r);

        draw_squares(level, x + r,
                y + r / 2 - new_r / 2, new_r);
    }
}
```

4. Strom – předloha



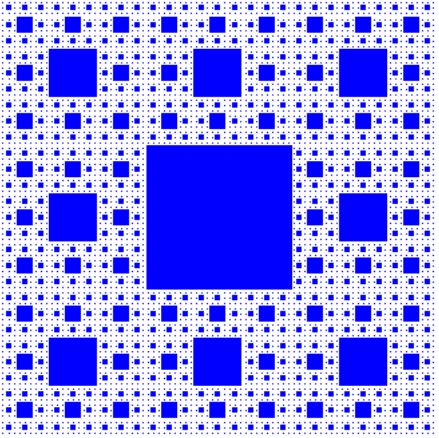
4. Strom – program 1./2

```
void draw_tree(int level, int x, int y,
         int size, int orient)
{
   int new_size = size / 1.732;   /* sqrt(3) */
   int new_x, new_y;

   /*
    * new_x = x - sin(orient) * size;
    * new_y = y - cos(orient) * size;
    */

   int orient_trans[2][8] = {
      /* X */ {  0,  1,  1,  1,  0, -1, -1, -1 },
      /* Y */ { -1, -1,  0,  1,  1,  1,  0, -1 }
   };
```

4. Strom – program 2./2

```
   new_x = x + orient_trans[X][orient] * size;
   new_y = y + orient_trans[Y][orient] * size;

   /* prostredni */
   printf("draw_line %d %d %d %d\n",
         x, y, new_x, new_y);

   /* vetve */
   if (level++ <= 7) {
      draw_tree(level, new_x, new_y,
            new_size, (orient + 7) % 8);
      draw_tree(level, new_x, new_y,
            new_size, (orient + 1) % 8);
   }
}
```

5. Sierpinského kobereček – předloha



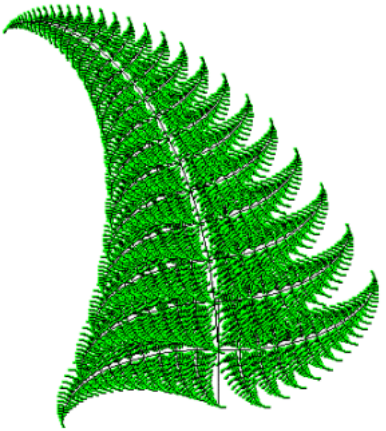
5. Sierpinského kobereček – program 1./2

```
void draw_squares(int level, int x, int y, int r) {
   float new_r = r / 3.;

   /* prostredni */
   printf("draw_rect_f %d %d %d %d\n",
         (int)(x + new_r), (int)(y + new_r),
         (int) new_r, (int) new_r);

   /* 3 okolo */
   if (level++ <= 4) {
      draw_squares(level, x + r * 0./3.,
            y + r * 0./3., new_r);
      draw_squares(level, x + r * 1./3.,
            y + r * 0./3., new_r);
      draw_squares(level, x + r * 2./3.,
            y + r * 0./3., new_r);
```

5. Sierpinského kobereček – program 2./2

```
        draw_squares(level, x + r * 0./3.,
                     y + r * 1./3., new_r);
        draw_squares(level, x + r * 2./3.,
                     y + r * 1./3., new_r);

        draw_squares(level, x + r * 0./3.,
                     y + r * 2./3., new_r);
        draw_squares(level, x + r * 1./3.,
                     y + r * 2./3., new_r);
        draw_squares(level, x + r * 2./3.,
                     y + r * 2./3., new_r);
    }
}
```

6. Kapradina – předloha



6. Kapradina – program 1./2

```
void draw_fern(int level, int x, int y,
      int size, float orient)
{
   float p;

   /* vypocet souradnic */
   int new_x  = x + cos(orient / 180. * M_PI) * size;
   int new_y  = y - sin(orient / 180. * M_PI) * size;

   /* prostredni */
   printf("draw_line %d %d %d %d\n",
         x, y, new_x, new_y);
```

6. Kapradina – program 2./2

```
   /* 3 okolo */
   if (level++ <= LEVEL_MAX) {
      draw_fern(level, new_x, new_y,
            size * 0.9, orient + 2);
      draw_fern(level, new_x, new_y,
            size * 0.4, orient - 80);
      draw_fern(level, new_x, new_y,
            size * 0.4, orient + 80);
   }
}
```

Konec