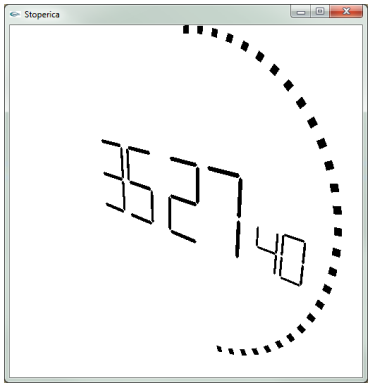
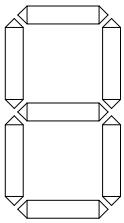
ELEKTROTEHNIČKI FAKULTET 17.06.2016.
Katedra za RTI

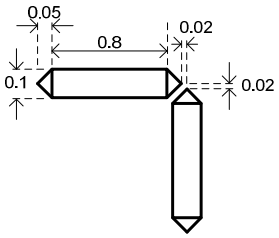
ISPIT IZ RAČUNARSKE GRAFIKE
praktični deo

1) [30] **OpenGL:** Napisati na jeziku C ili C++ deo programa za crtanje scene prikazane na slici 1a primenom grafičke biblioteke OpenGL. Scena se sastoji od brojčanika štoperice i indikatora sekundi proteklih u tekućoj minuti čiji su elementi kružno raspoređeni oko brojčanika. Brojčanik prikazuje broj proteklih minuta, sekundi i stotih delova sekunde (35 minuta, 27 sekundi i 40 stotih delova sekunde na slici 1a). Cifre brojčanika su formirane od sedmosegmentnih elemenata čiji je oblik prikazan na slici 1b. Relativne dimenzije i rastojanja segmenata prikazani su na slici 1c. Elementi scene se crtaju crnom bojom na beloj podlozi. Koristi se projekcija sa perspektivom. Napisati posebnu funkciju koja vrši osnovnu inicijalizaciju OpenGL sistema potrebnu za crtanje scene. Parametre projekcije podesiti i posmatračku kameru približno postaviti u položaj koji bi proizveo sliku 1a. Na raspolaganju je glavni program koji inicijalizuje biblioteku GLUT i otvara prozor. U funkciji koja crta sadržaj scene realizovano je merenje proteklog vremena koje prikazuje štoperica, izraženo u milisekundama, i njegovo smeštanje u promenljivu `proteklo_msek`.



Slika 1a


Slika 1b



Slika 1c

2) [20] Korišćenjem Bresenham-ovog algoritma za crtanje kružnice kao osnove, potprograma za crtanje linije `Line(xa, ya, xb, yb)` koji crta liniju od tačke (x_a, y_a) do tačke (x_b, y_b) i potprograma za crtanje tačke `Plot(x, y)` koji crta tačku (x, y), napisati potprogram čija je deklaracija

void Crtaj(int x1, int y1, int x2, int y2, bool mod);


a) b)
Slika 2

gde *mod* određuje da li će se crtati figura 2a ili figura 2b, na slici 2. *mod* je *true* za figuru 2b. Tačka (x1, y1) predstavlja donji levi, a (x2, y2) gornji desni ugao figure. Pretpostaviti da je pre poziva potprograma izvršena provera valjanosti datih koordinata. Crtanje je crnom olovkom na beloj pozadini. Skelet potprograma se nalazi u fajlu **Resenje.cpp**.

Napomene: 1. Praktični deo ispita, odnosno izrada zadataka 1 i 2, traje 160 minuta.
2. Rešenje zadataka 1 i 2 se predaje u obliku **.cpp** fajla u predviđenom folderu na računaru.
3. Nije dozvoljeno uz sebe imati mobilni telefon, bez obzira da li je uključen ili isključen.
4. Dozvoljena je upotreba proizvoljne literature.

Rešenja zadataka

Jun 2016

1) Rešenje (delovi programa obeleženi sivo nisu deo očekivanog rešenja)

```
#include <Windows.h>
#include <stdlib.h>
#include <time.h>
#include <GL/glut.h>
#include <GL/GL.H>
void promenaProzora(int width, int height) {
  glViewport(0, 0, width, height);
  glMatrixMode(GL_PROJECTION);
  glLoadIdentity();
  gluPerspective(45,(float)width/height,
                 1, 20);
  glMatrixMode(GL_MODELVIEW);
}

void crtajSegment() {
  glBegin(GL_QUADS);
  glVertex2f(-0.4f, -0.05f);
  glVertex2f(0.4f, -0.05f);
  glVertex2f(0.4f, 0.05f);
  glVertex2f(-0.4f, 0.05f);
  glEnd();
  glBegin(GL_TRIANGLES);
  glVertex2f(-0.45f, 0.f);
  glVertex2f(-0.4f, -0.05f);
  glVertex2f(-0.4f, 0.05f);
  glVertex2f(0.45f, 0.f);
  glVertex2f(0.4f, 0.05f);
  glVertex2f(0.4f, -0.05f);
  glEnd();
}

int dohvatiMasku(int cifra){
  switch (cifra)  {
    case 0: return 0x77; break;
    case 1: return 0x14; break;
    case 2: return 0x6E; break;
    case 3: return 0x3E; break;
    case 4: return 0x1D; break;
    case 5: return 0x3B; break;
    case 6: return 0x7B; break;
    case 7: return 0x16; break;
    case 8: return 0x7F; break;
    case 9: return 0x3F; break;
  }
  return 0;
}

void crtajCifru(const int maska) {
  if (maska & 1) {
    glPushMatrix();
    glTranslatef(-0.47f, 0.47f, 0.0f);
    glRotatef(90, 0, 0, 1);
    crtajSegment(); glPopMatrix();
  }
  if (maska & 2) {
    glPushMatrix();
    glTranslatef(0.0f, 0.94f, 0.0f);
    crtajSegment(); glPopMatrix();
  }
  if (maska & 4) {
    glPushMatrix();
    glTranslatef(0.47f, 0.47f, 0.0f);
    glRotatef(90, 0, 0, 1);
    crtajSegment(); glPopMatrix();
  }
  if (maska & 8) { crtajSegment(); }
  if (maska & 16) {
    glPushMatrix();
    glTranslatef(0.47f, -0.47f, 0.0f);
    glRotatef(90, 0, 0, 1);
    crtajSegment(); glPopMatrix();
  }
  if (maska & 32) {
    glPushMatrix();
    glTranslatef(0.0f, -0.94f, 0.0f);
    crtajSegment(); glPopMatrix();
  }
  if (maska & 64) {
    glPushMatrix();
    glTranslatef(-0.47f, -0.47f, 0.0f);
    glRotatef(90, 0, 0, 1);
    crtajSegment(); glPopMatrix();
  }
}

void crtajStopericu(long msek) {
  int min = msek / 60000;
  int sek = (msek / 1000) % 60;
  glPushMatrix();
  glColor3f(0,0,0);
  glTranslatef(-2.9f, 0, 0);
  crtajCifru(dohvatiMasku((min/10)%10));
  glTranslatef(1.2f, 0, 0);
  crtajCifru(dohvatiMasku((min) % 10));
  glTranslatef(1.6f, 0, 0);
  crtajCifru(dohvatiMasku((sek/10)%10));
  glTranslatef(1.2f, 0, 0);
  crtajCifru(dohvatiMasku((sek) % 10));
  glTranslatef(1.2f, -0.47, 0);
  glScalef(0.5f, 0.5f, 1);
  crtajCifru(dohvatiMasku((msek/100) % 10));
  glTranslatef(1.2f, 0, 0);
  crtajCifru(dohvatiMasku((msek/10) % 10));
  glPopMatrix();
}

void crtajOznake(long msek) {
  glPushMatrix();
  glScalef(7.5f, 7.5f, 1.0f);
  int sekundi = (msek / 1000) % 60;
  for (int i = 0; i <= sekundi; i++) {
    glPushMatrix();
    glRotatef(-i * 6, 0, 0, 1);
    glTranslatef(0, 0.5f, 0);
    glBegin(GL_QUADS);
    glVertex2f(-0.01f, -0.01f);
    glVertex2f(0.01f, -0.01f);
    glVertex2f(0.01f, 0.01f);
    glVertex2f(-0.01f, 0.01f);
    glEnd();  glPopMatrix();
  } glPopMatrix();
}

void crtajScenu(void) {
  static long prethodno_vreme = 0;
  static long proteklo_msek = 0;
  long proteklo = 0;
  if (prethodno_vreme != 0) {
    proteklo = timeGetTime() -
               prethodno_vreme;
    if (proteklo < 40) {
      Sleep(40 - proteklo);
      glFinish();
      return;
    }
    proteklo_msek += proteklo;
    prethodno_vreme += proteklo;
  }
  else
    prethodno_vreme = timeGetTime();
 glClear(GL_COLOR_BUFFER_BIT |
         GL_DEPTH_BUFFER_BIT);
 glLoadIdentity();
 glTranslatef(0, 0, -10.0f);
 glRotatef(20, 1, 0, 0);
 glRotatef(-45, 0, 1, 0);
 crtajStopericu(proteklo_msek);
 crtajOznake(proteklo_msek);
 glFlush();
}

void init() {
  glClearColor (1.0, 1.0, 1.0, 0.0);
  glEnable(GL_DEPTH_TEST);
  glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);
  glMatrixMode(GL_MODELVIEW);
  glLoadIdentity();
}

int main(int argc, char **argv) {
 glutInit(&argc, argv);
 glutInitDisplayMode (GLUT_RGB | GLUT_DEPTH);
 glutInitWindowSize(500, 500);
 glutInitWindowPosition(100, 100);
 glutCreateWindow("Stoperica");
 init();
 glutDisplayFunc(crtajScenu);
 glutReshapeFunc(promenaProzora);
 glutIdleFunc(crtajScenu);
 glutMainLoop();
 return 0; }
```

2)

```
void crtaj(int x1, int y1, int x2, int y2, bool mod) {
  line(x1, y1, x2, y1);
  line(x2, y1, x2, y2);
  line(x2, y2, x1, y2);
  line(x1, y2, x1, y1);

  int xc1 = (x1+x2)/2;
  int yc1 = (y1+y2)/2;
  int r1 = (x2-x1)/2;
  int r2 = r1/2;
  int xc2 = xc1+r2;
  int x = r1; int y=0;
  int d = 3-2*x;

  while(x>=y)  {
        if( mod) {
          line(xc1-x, yc1-y, xc1+x, yc1-y);
          line(xc1-y, yc1-x, xc1+y, yc1-x);
        }
        else  {
          line(x1, yc1-y, xc1-x, yc1-y);
          line(x1, yc1-x, xc1-y, yc1-x);
          line(xc1+x, yc1-y, x2, yc1-y);
          line(xc1+y, yc1-x, x2, yc1-x);
        }
        if( d < 0 )
          d += 4*y+6;
        else {
          d += 4 * (y-x)+10;
          x--;
        }
        y++;
  }
  x = r2;
  y = 0;
  d = 3-2*x;
  while(x >= y)  {
    if( mod )  {
        line(xc2-x, yc1+y, xc2+x, yc1+y);
        line(xc2-y, yc1+x, xc2+y, yc1+x);
    }
    else {
        line(x1, yc1+y, xc2-x, yc1+y);
        line(xc2+x, yc1+y, x2, yc1+y);

        line(x1, yc1+x, xc2-y, yc1+x);
        line(xc2+y, yc1+x, x2, yc1+x);
        line(x1, yc1+r2+y, x2, yc1+r2+y);
        line(x1, y2-y, x2, y2-y);
    }

    if( d < 0 )
        d += 4*y+6;
    else {
        d += 4 * (y-x)+10;
        x--;
    }
    y++;
  }
}
```