UNIVERSITI SAINS MALAYSIA

First Semester Examination
2011/2012 Academic Session

January 2012

# MSG 387 – Computer Graphics
## *[Grafik Komputer]*

Duration : 3 hours
*[Masa : 3 jam]*

Please check that this examination paper consists of SIXTEEN pages of printed material before you begin the examination.

*[Sila pastikan bahawa kertas peperiksaan ini mengandungi ENAM BELAS muka surat yang bercetak sebelum anda memulakan peperiksaan ini.]*

**Instructions** : Answer **all six** [6] questions.

*[**Arahan** : Jawab **semua enam** [6] soalan.*

**The question papers shall not be taken out from the examination hall and will be collected by invigilators.**

***Kertas soalan ini tidak boleh dibawa keluar daripada dewan peperiksaan dan akan dikutip oleh pengawas peperiksaan.***

In the event of any discrepancies, the English version shall be used.

*[Sekiranya terdapat sebarang percanggahan pada soalan peperiksaan, versi Bahasa Inggeris hendaklah diguna pakai].*

1. Explain the following definitions,
   (a) Frame buffer;
   (b) Pixel Depth;
   (c) Scan line;
   (d) Interlace scanning;
   (e) RGB and CMY as additive and subtractive color models respectively.

[10 marks]

*1. Jelaskan definisi-definisi berikut,*
   *(a) Penimbal kerangka;*
   *(b) Kedalaman piksel;*
   *(c) Garis imbas;*
   *(d) Pengimbasan selang-seli;*
   *(e) RGB dan CMY sebagai penambahan dan pengurangan model warna masing-masing.*

*[10 markah]*

2. Given a raster screen with resolution 1280 by 1024 pixels. Pixel depth is 16 bits. Refresh rate is 60 frames per second (fps), with horizontal retrace time of 5 microseconds and vertical retrace time of 500 microseconds,
   (a) What is the aspect ratio of the screen? Leave your answer in fraction.
   (b) What is the total number of distinct colors which can be displayed on the screen?
   (c) What is the frame buffer size (in KB) required?
   (d) How many pixels can be transferred per second?
   (e) What is the access time per pixel?
   (f) What is the total refresh time per frame spent in retrace of the electron beam for a non-interlaced raster system?

[10 marks]

*2. Diberi suatu skrin raster dengan leraian 1280 darab 1024. Kedalaman piksel ialah 16 bit. Kadar segar semula ialah 60 kerangka per saat (fps), dengan tempoh surih balik mendatar 5 mikro-saat dan tempoh surih balik menegak 500 mikro-saat.*
   *(a) Apakah nisbah bidang untuk skrin tersebut? Tinggalkan jawapan anda dalam pecahan.*
   *(b) Apakah jumlah bilangan warna yang berbeza yang dapat dipaparkan di atas skrin?*
   *(c) Apakah saiz penimbal kerangka (dalam KB) yang diperlukan?*
   *(d) Berapakah piksel yang dapat dipindahkan dalam satu saat?*
   *(e) Apakah masa capaian per piksel?*
   *(f) Apakah jumlah masa segar semula per kerangka yang digunakan dalam surih balik untuk pancaran electron bagi suatu sistem raster yang bukan selang-seli?*

*[10 markah]*

3. Find the plane equation defined by (2, 10, 4), (10, 2, 10) and (4, 10, 8). Then find the shortest distance between this plane to point (20, 10, 20).

[5 marks]

*3. Cari persamaan satah yang didefinisikan oleh (2, 10, 4), (10, 2, 10) dan (4, 10, 8). Kemudian cari jarak terpendek di antara satah ini dengan titik (20, 10, 20).*

*[5 markah]*

4. (a) Explain how Bresenham and Midpoint algorithms for straight line are derived for the first octant in Cartesian coordinate system, i.e $0 < m < 1$, where $m$ denotes the slope of a straight line.
   (b) Which algorithm is more suitable for circle drawing? Why?
   (c) Derive DDA (Digital Differential Analyzer) line drawing algorithms for the first quadrant in Cartesian coordinate system, i.e. $m > 0$, where $m$ is the slope for a straight line.
   (d) Generate a straight line from (3, 10) to (6,16) by using Midpoint algorithm and DDA algorithm.
   (e) From the result of question 4(d), conclude which algorithm is better and explain the reason.

[25 marks]

*4. (a) Jelaskan bagaimana algoritma Bresenham dan Midpoint untuk garis lurus boleh diterbitkan dari oktan pertama dalam sistem koordinat Cartesian, i.e. $0 < m < 1$, di mana m ialah kecerunan untuk suatu garis lurus.*
   *(b) Algoritma yang mana satu adalah lebih sesuai untuk melukis suatu bulatan? Kenapa?*
   *(c) Terbitkan algoritma-algoritma untuk melukis garis lurus dalam DDA (Digital Differential Analyzer) untuk kuadran pertama dalam sistem koordinat Cartesian, i.e. $m > 0$, di mana m ialah kecerunan garis lurus.*
   *(d) Janakan satu garis lurus dari (3, 10) ke (6,16) dengan menggunakan algoritma-algoritma Midpoint dan DDA.*
   *(e) Dari jawapan soalan 4(d), buat kesimpulan bahawa algoritma manakah lebih baik dan jelaskan.*

*[25 markah]*

5. (a) Write transformation matrices for 3 dimensional translation, as well as rotations about *x*-, *y*- and *z*-axes.

(b) Write respective inverse matrices for the transformation matrices in 5(a).

(c) Derive the transformation matrices for rotation about an arbitrary line, **L**, which is not parallel to Cartesian coordinate axes. Explain in detail how each transformation matrix is obtained. Your explanations should be assisted by appropriate figures. You may follow the guidelines given below:

Step 1. Translate **L** so that it passes through the center of origin.
Step 2. Rotate **L** so that it overlaps on *z*-axis.
Step 3. Rotate the object about *z*-axis.
Step 4. Rotate **L** so that it is the reverse of Step 2.
Step 5. Translate **L** back to its original position and orientation.

(d) A space point **P**(2,3,6) is rotated 30° counter-clockwise about a space line, **L**. Direction of **L** is given as (5, 4, 3) and passes through point **A**(1, 1, 1). Find the new coordinate for **P** after the rotation.

[25 marks]

5. *(a) Tuliskan matriks-matriks transformasi 3 dimensi untuk translasi, dan juga untuk putaran sekitar paksi-paksi x, y dan z.*

*(b) Tuliskan matriks-matriks songsang untuk matriks-matriks transformasi di soalan 5(a).*

*(c) Terbitkan matriks-matriks transformasi bagi putaran sekitar garis sebarangan **L**, yang tidak selari dengan paksi-paksi koordinat Cartesian. Jelaskan dengan mendalam bagaiman setiap matriks transformasi didapati. Penjelasan anda mestilah dibantu dengan gambarajah yang bersesuaian. Anda boleh mengikut garis panduan yang diberi seperti berikut:*

*Langkah 1. Translasikan **L** supaya **L** melalui asalan.*
*Langkah 2. Putarkan **L** supaya **L** bertindih dengan paksi z.*
*Langkah 3. Putarkan objek sekitar paksi z.*
*Langkah 4. Putarkan **L** supaya **L** adalah songsang untuk Langkah 2.*
*Langkah 5. Translasikan **L** balik ke posisi dan orientasi asalnya.*

*(d) Suatu titik ruang **P**(2,3,6) diputarkan sebanyak 30° arah songsang jam sekitar suatu garis ruang, **L**. Arah **L** diberi sebagai (5, 4, 3) dan melalui titik **A**(1, 1, 1). Cari koordinat baru **P** selepas putaran tersebut.*

*[25 markah]*

6. (a) Given the coordinate points from $\mathbf{P}_1$ to $\mathbf{P}_{10}$ with locations as displayed in Figure 1. What are the polygons produced by the OpenGL code segments from (i) to (iii)? Draw lines to display the polygons according to the modes of the `glBegin` function below. Draw your answers on page 15 and attach this page to your answer booklet.

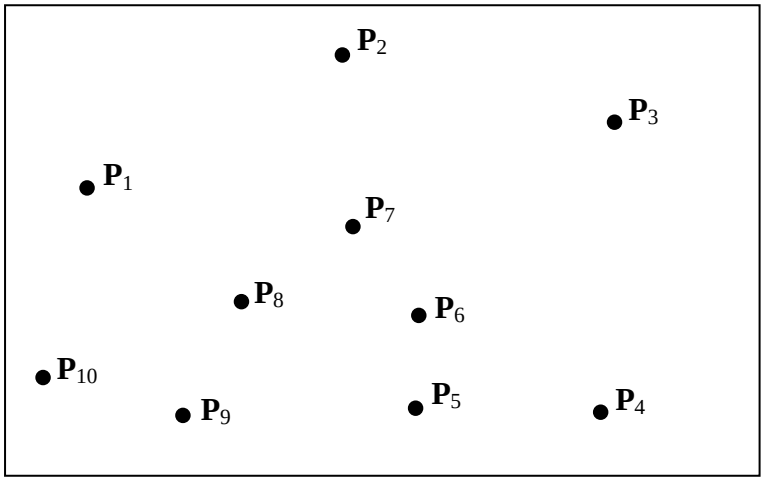



**Figure 1.**

(i)

```
glBegin(GL_POLYGON);
   glVertex2iv (P1);
   glVertex2iv (P2);
   glVertex2iv (P3);
   glVertex2iv (P4);
   glVertex2iv (P5);
   glVertex2iv (P6);
   glVertex2iv (P7);
   glVertex2iv (P8);
   glVertex2iv (P9);
   glVertex2iv (P10);
glEnd();
```

(ii)

```
glBegin(GL_QUADS);
   glVertex2iv (P1);
   glVertex2iv (P2);
   glVertex2iv (P3);
   glVertex2iv (P4);
   glVertex2iv (P5);
   glVertex2iv (P6);
   glVertex2iv (P7);
   glVertex2iv (P8);
   glVertex2iv (P9);
   glVertex2iv (P10);
glEnd();
```

(iii)

```
glBegin(GL_QUAD_STRIP);
   glVertex2iv (P10);
   glVertex2iv (P9);
   glVertex2iv (P1);
   glVertex2iv (P8);
   glVertex2iv (P2);
   glVertex2iv (P7);
   glVertex2iv (P3);
   glVertex2iv (P6);
   glVertex2iv (P4);
   glVertex2iv (P5);
glEnd();
```

(b) Let $\mathbf{P}_1$=(0, 0, –50), $\mathbf{P}_2$=(50, 0, –50), $\mathbf{P}_3$=(50, 50, –50), $\mathbf{P}_4$=(0, 50, –50), $\mathbf{P}_5$=(50, 50, 0), $\mathbf{P}_6$=(50, 0, 0), $\mathbf{P}_7$=(0, 50, 0) and $\mathbf{P}_8$=(0, 0, 0). A cube is formed by the eight vertices from $\mathbf{P}_1$ to $\mathbf{P}_8$ as in Figure 2.

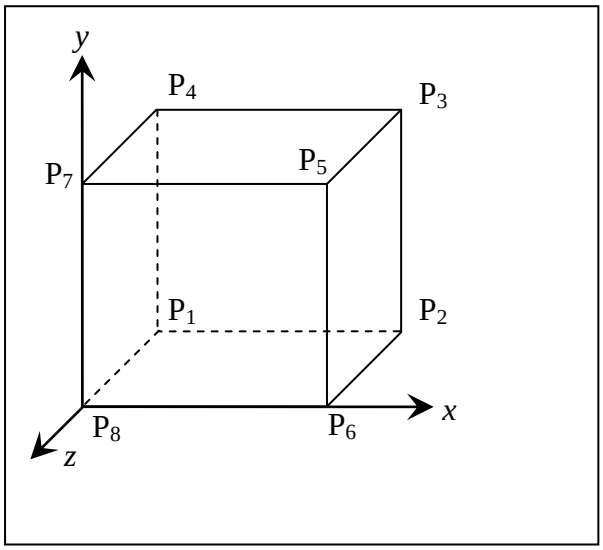



**Figure 2.**

Consider the OpenGL segment on the next page which draws the cube by using `glDrawElements`. What are the values to be initialized for the array `vertIndex`? The syntax and usage for `glVertexPointer` and `glDrawElements` are given in the Appendix.

```
:
GLint vertices [8][3] = {      // declaration of vertices
 {  0,   0, -50},   // P1
 { 50,   0, -50},   // P2
 { 50,  50, -50},   // P3
 {  0,  50, -50},   // P4
 { 50,  50,   0},   // P5
 { 50,   0,   0},   // P6
 {  0,  50,   0},   // P7
 {  0,   0,   0}};  // P8

glEnableClientState(GL_VERTEX_ARRAY);
glVertexPointer (3, GL_INT, 0, vertices);
GLubyte vertIndex[] = {
      // what are the values to be initialized here?
};
:
glDrawElements(GL_QUADS, 24, GL_UNSIGNED_BYTE, verIndex);
:
```

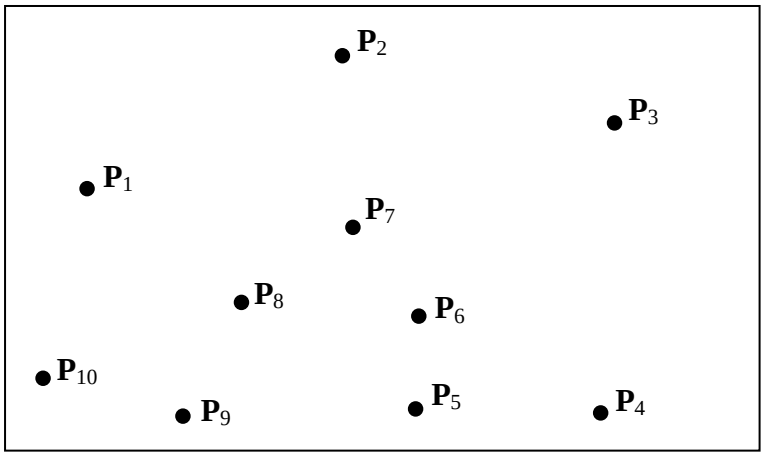
(c) Consider the following OpenGL program segment. Draw your outputs on the answer sheet provided in page 16. The first triangle in black has already been drawn for you.

```
:
void draw_triangle(void)
{
   glBegin (GL_LINE_LOOP);
      glVertex2f( 3.0f,  5.0f);
      glVertex2f(10.0f, 10.0f);
      glVertex2f( 1.0f, 14.0f);
   glEnd();
}

void display(void)
{
   glClear (GL_COLOR_BUFFER_BIT);

   glColor3f (0.0, 0.0, 0.0);
   glLineWidth(2);
   glLoadIdentity ();
   draw_triangle (); // BLACK triangle
   draw_grid();
   drawAxis(0,0,0);
   glTranslatef (5.0, 9.0, 0.0);
   glRotatef(90.0, 0,0,1);
   glTranslatef (-5.0, -9.0, 0.0);
   glColor3f (1.0, 0.0, 0.0);
```

```
   draw_triangle (); // RED triangle
   glTranslatef (3.0, 2.0, 0.0);
   glScalef(2.0, 0.5, 1.0);
   glTranslatef (-3.0, -2.0, 0.0);
   glColor3f (0.0, 1.0, 0.0);
   draw_triangle (); // GREEN triangle
   glTranslatef (-7.0, 0.0, 0.0);
   glColor3f (0.0, 0.0, 1.0);
   draw_triangle (); // BLUE triangle

   glFlush ();
}
:
```

[25 marks]

6. (a) *Diberi titik-titik koordinat dari* $\mathbf{P}_1$ *ke* $\mathbf{P}_{10}$ *dengan lokasi-lokasi seperti yang tunjukkan di Gambarajah 1. Apakah poligon-poligon yang dihasilkan oleh segmen-segmen OpenGL dari (i) ke (iii)? Lukis garis untuk memaparkan poligon-poligon berdasarkan mod-mod fungsi* `glBegin` *di bawah. Lukiskan jawapan anda di mukasurat 15 dan sertakan mukasurat tersebut ke buku jawapan anda.*




***Gambarajah 1.***

*(i)*

```
glBegin(GL_POLYGON);
   glVertex2iv (P1);
   glVertex2iv (P2);
   glVertex2iv (P3);
   glVertex2iv (P4);
   glVertex2iv (P5);
   glVertex2iv (P6);
   glVertex2iv (P7);
   glVertex2iv (P8);
   glVertex2iv (P9);
   glVertex2iv (P10);
glEnd();
```

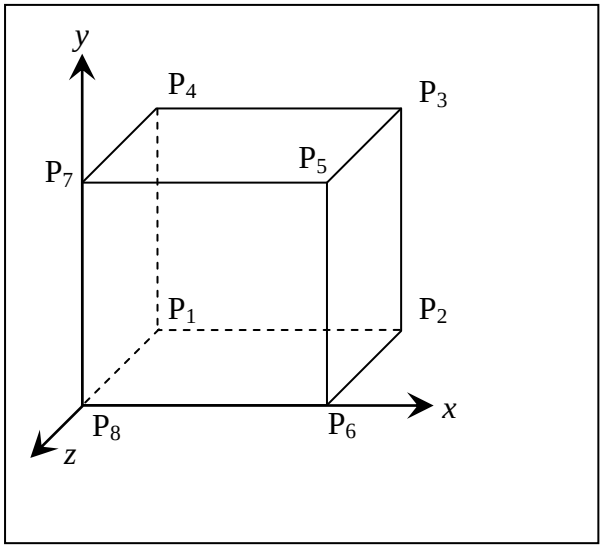
*(ii)*

```
glBegin(GL_QUADS);
   glVertex2iv (P1);
   glVertex2iv (P2);
   glVertex2iv (P3);
   glVertex2iv (P4);
   glVertex2iv (P5);
   glVertex2iv (P6);
   glVertex2iv (P7);
   glVertex2iv (P8);
   glVertex2iv (P9);
   glVertex2iv (P10);
glEnd();
```

*(iii)*

```
glBegin(GL_QUAD_STRIP);
   glVertex2iv (P10);
   glVertex2iv (P9);
   glVertex2iv (P1);
   glVertex2iv (P8);
   glVertex2iv (P2);
   glVertex2iv (P7);
   glVertex2iv (P3);
   glVertex2iv (P6);
   glVertex2iv (P4);
   glVertex2iv (P5);
glEnd();
```

(b) *Biar* $\boldsymbol{P}_1$*=(0, 0, –50),* $\boldsymbol{P}_2$*=(50, 0, –50),* $\boldsymbol{P}_3$*=(50, 50, –50),* $\boldsymbol{P}_4$*=(0, 50, –50),* $\boldsymbol{P}_5$*=(50, 50, 0),* $\boldsymbol{P}_6$*=(50, 0, 0),* $\boldsymbol{P}_7$*=(0, 50, 0) dan* $\boldsymbol{P}_8$*=(0, 0, 0). Suatu kuib dibentukkan oleh lapan mercu dari* $\boldsymbol{P}_1$ *to* $\boldsymbol{P}_8$ *seperti yang ditunjukkan di Gambarajah 2.*




***Gambarajah 2.***

*Pertimbangkan segment OpenGL berikut yang melukis kiub tersebut dengan menggunakan fungsi* `glDrawElements`*. Apakah nilai-nilai awal yang patut diberi kepada tatasusunan* `vertIndex`*? Sintaks dan kegunaan fungsi-fungsi* `glVertexPointer` *dan* `glDrawElements` *diberi di Lampiran.*

```
:
GLint vertices [8][3] = {      // pengisytiharan mercu-mercu
 {  0,   0, -50},   // P1
 { 50,   0, -50},   // P2
 { 50,  50, -50},   // P3
 {  0,  50, -50},   // P4
 { 50,  50,   0},   // P5
 { 50,   0,   0},   // P6
 {  0,  50,   0},   // P7
 {  0,   0,   0}};  // P8

glEnableClientState(GL_VERTEX_ARRAY);
glVertexPointer (3, GL_INT, 0, vertices);
GLubyte vertIndex[] = {
      // apakah nilai-nilai pengawalan yang patut diberi di sini?
};
:
glDrawElements(GL_QUADS, 24, GL_UNSIGNED_BYTE, verIndex);
:
```

*(c) Pertimbangkan segmen program OpenGL berikut. Lukiskan output-output anda di atas kertas jawapan yang disediakan di mukasurat 16. Segitiga pertama yang berwarna hitam telah dilukis untuk anda.*

```
:
void draw_triangle(void)
{
   glBegin (GL_LINE_LOOP);
      glVertex2f( 3.0f,  5.0f);
      glVertex2f(10.0f, 10.0f);
      glVertex2f( 1.0f, 14.0f);
   glEnd();
}

void display(void)
{
   glClear (GL_COLOR_BUFFER_BIT);

   glColor3f (0.0, 0.0, 0.0);
   glLineWidth(2);
```

```
   glLoadIdentity ();
    draw_triangle (); // segitiga HITAM
   draw_grid();
   drawAxis(0,0,0);
   glTranslatef (5.0, 9.0, 0.0);
   glRotatef(90.0, 0,0,1);
   glTranslatef (-5.0, -9.0, 0.0);
   glColor3f (1.0, 0.0, 0.0);
   draw_triangle (); // segitiga MERAH
   glTranslatef (3.0, 2.0, 0.0);
   glScalef(2.0, 0.5, 1.0);
   glTranslatef (-3.0, -2.0, 0.0);
   glColor3f (0.0, 1.0, 0.0);
   draw_triangle (); // segitiga HIJAU
   glTranslatef (-7.0, 0.0, 0.0);
   glColor3f (0.0, 0.0, 1.0);
   draw_triangle (); // segitiga BIRU

   glFlush ();
}
:
```

*[25 markah]*

**APPENDIX**

```
void glVertexPointer(GLint         size,
                     GLenum        type,
                     GLsizei       stride,
                     const GLvoid *pointer);
```

**Parameters**

*size*

Specifies the number of coordinates per vertex. Must be 2, 3, or 4. The initial value is 4.

*type*

Specifies the data type of each coordinate in the array. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

*stride*

Specifies the byte offset between consecutive vertices. If *stride* is 0, the vertices are understood to be tightly packed in the array. The initial value is 0.

*pointer*

Specifies a pointer to the first coordinate of the first vertex in the array. The initial value is 0.

```
void glDrawElements(GLenum        mode,
                    GLsizei       count,
                    GLenum        type,
                    const GLvoid *indices);
```

**Parameters**

*mode*

Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

*count*

Specifies the number of elements to be rendered.

*type*

Specifies the type of the values in *indices*. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

*indices*

Specifies a pointer to the location where the indices are stored.

…13/-

***LAMPIRAN***

```
void glVertexPointer(GLint          size,
                     GLenum         type,
                     GLsizei        stride,
                     const GLvoid *pointer);
```

***Parameters***

`size`

*Menyatakan bilangan koordinat per mercu. Mestilah bernilai 2, 3 atau 4. Nilai awalan ialah 4.*

`type`

*Menyatakan jenis data bagi setiap koordinat dalam tatasusunan. Pemalar simbolik* `GL_SHORT`*,* `GL_INT`*,* `GL_FLOAT`*, atau* `GL_DOUBLE` *boleh diterima. Nilai awalan ialah* `GL_FLOAT`*.*

`stride`

*Menyetakan offset bait di antara mercu-mercu yang berturutan. Jika* `stride` *ialah 0, mercu-mercu dianggap adalah disusun dengan padat. Nilai awalan ialah 0.*

`pointer`

*Menyatakan suatu penuding kepada koordinat pertama bagi mercu pertama dalam tatasusunan. Nilai awalan ialah 0.*

```
void glDrawElements(GLenum         mode,
                    GLsizei        count,
                    GLenum         type,
                    const GLvoid *indices);
```

***Parameters***

`mode`

*Menyatakan jenis-jenis primitive yang dibalas. Pemalar simbolik* `GL_POINTS`*,* `GL_LINE_STRIP`*,* `GL_LINE_LOOP`*,* `GL_LINES`*,* `GL_TRIANGLE_STRIP`*,* `GL_TRIANGLE_FAN`*,* `GL_TRIANGLES`*,* `GL_QUAD_STRIP`*,* `GL_QUADS`*, dan* `GL_POLYGON` *boleh diterima.*

`count`

*Menyatakan bilangan unsure yang akan dibalas.*

`type`

*Menyatakan jenis-jenis nilai dalam* `indices`*. Mestilah salah satu daripada* `GL_UNSIGNED_BYTE`*,* `GL_UNSIGNED_SHORT`*, or* `GL_UNSIGNED_INT`*.*

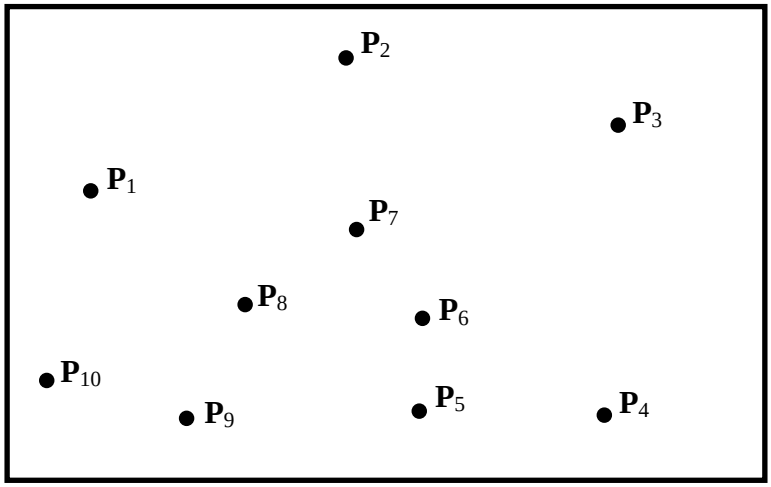
`indices`

*Menyatakan satu penuding kepada lokasi di mana mercu-mercu disimpan.*

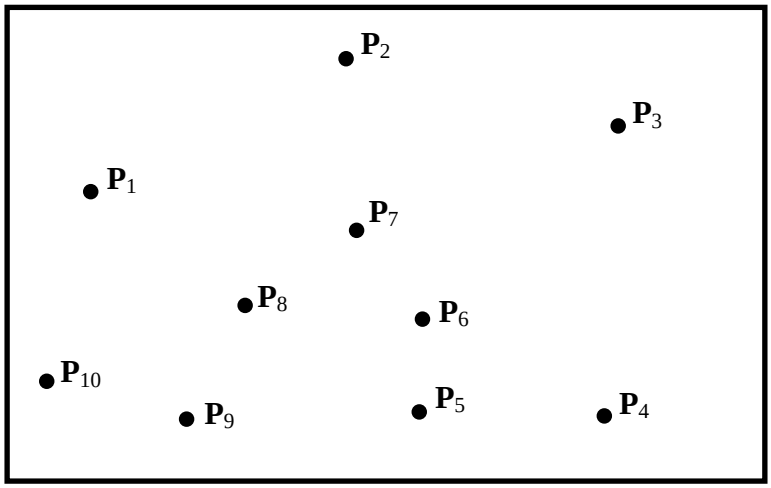
**THIS PAGE IS LEFT BLANK INTENTIONALLY**
***MUKASURAT INI DIKOSONGKAN DENGAN SENGAJA***

**Answer Sheet for Question 6 (a) / *Kertas Jawapan untuk Soalan 6 (a)***

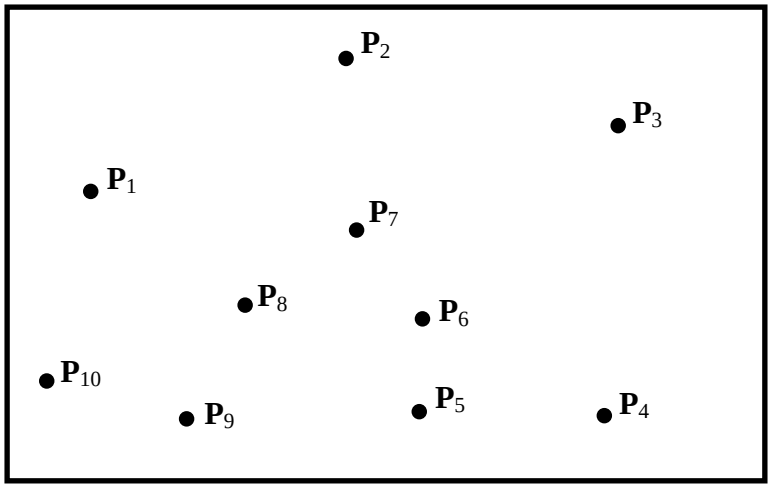
(i)




(ii)




(iii)



…16/-

**Answer Sheet for Question 6 (c) /** ***Kertas Jawapan untuk Soalan 6 (c)***