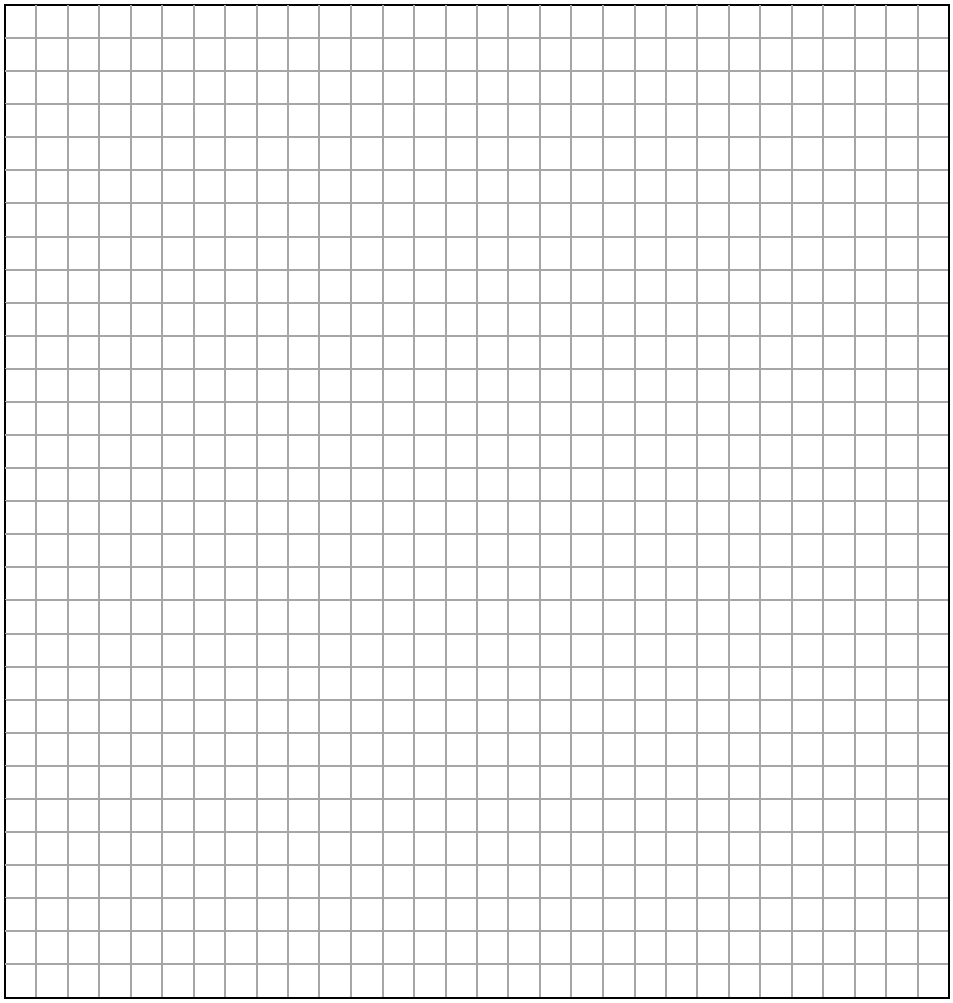
https://dystosvita.gnomio.com/

https://www.open**processing.org**/sketch/590044



size(x,y); // розмір

background(r,g,b); // колір тла

stroke (r,g,b,a); //колір ліній

fill(r,g,b,a); // колір заливки

noFill(); // без заливки

strokeWeight(a); // товщина ліній

line(x1,y1,x2,y2); // лінія

rect(x,y,a,b); // прямокутник

quad(x1, y1, x2, y2, x3, y3, x4, y4) ; // багатокутник

ellipse(x,y,a,b); // еліпс

arc(x, y, a, b, start, stop, mode); // сегмент, півколо, хорда

PIE, OPEN, CHORD

Змінні – значення, які будуть мінятись у програмі. Потрібно оголосити тип змінної.

int x,a,y;

Випадкове число – набуває дійсне значення у діапазоні від 0 до вказаного в дужках

x=int(random(500)+100);

```
size(x,y);
background(r,g,b);
int red=int(random(255));
int green=int(random(255));
int blue=int(random(255));
stroke(red,green,blue);
strokeWeight(a);
line(x1,y1,x2,y2);
```

	ПочатокХ	ПочатокУ	КінецьХ	КінецьУ
1 лінія				
2 лінія				
3 лінія				
Закономірність				

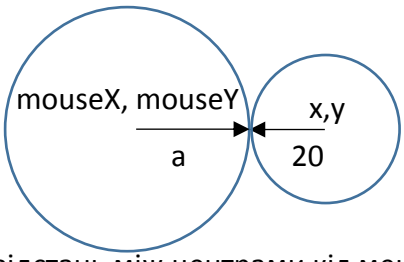
Цикли

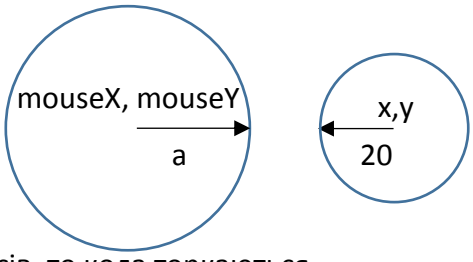
```
while (x<500) {
 x+=5;
line(x,0,x,300);
}
```

	ПочатокХ	ПочатокУ	КінецьХ	КінецьУ
1 лінія				
2 лінія				
3 лінія				
Закономірність				

```
for {int x=0; x<300; x=x+10}
(line(x,0,x,300);)
```

Виявлення торкання – потрібно порівняти координати двох кіл. Одне з них – має центр в x,y та розмір a, друге – має центр в mouseX,mouseY і розмір 20.





Якщо відстань між центрами кіл менша, ніж сума їх радіусів, то кола торкаються.

Відстань між центрами кіл **dist**(mouseX, mouseY, x,y) Сума радіусів кіл: a+20

Якщо відстань між центрами кіл більша, ніж сума їх радіусів, то кола торкаються.

if (dist(mouseX, mouseY, x,y) <a+20) { }

Процедури проекту

```
void setup(){
        size(700,700);
        background(120, 120, 120);
}
void draw(){
        background(120,120,120);
        ellipse(mouseX,mouseY,50,50);
}
```

Побудови по колу

```
size(500,500);
int n=36;  //кількість поворотів, або сторін
float f=0 ; //початковий кут
float df= 2*PI/n  ; //кут повороту такий, щоб за n кроків пройти 2π
int r=200; // радіус
strokeWeight(6);
for (int promin=0; promin<n; promin++)  //для кожного з n променів
{
   int x= 250+round(50*cos(f)); //обчислюємо координату x за рівнянням кола
   int y= 250+round(50*sin(f)); //обчислюємо координату y за рівнянням кола
   point (x, y); //малюємо точку x,y
   f += df; //змінюємо кут f на значення повороту
}
```

Обчислювальне мислення

☐ Логіка	☐ Креативність
☐ Алгоритм	☐ Наполегливість
☐ Декомпозиція (розподіл на частини)	☐ Експерименти
☐ Об’єднання в групи	☐ Помилки
☐ Абстракція	☐ Співпраця
☐ Шаблон	☐ Оцінювання та вибір

Об’єкти та класи

```
// Клас Автомобіль
class Car {
color c;
float xpos, ypos, xspeed;
// Конструктор об'єкта (опис властивостей)
Car (color tempC, float tempXpos, float tempYpos, float tempXspeed)  {
c = tempC;
xpos = tempXpos;
ypos = tempYpos;
xspeed = tempXspeed;
}
// Функціонал (методи)
 void display() {                    // Показати
   stroke(0);
   fill(c);
   rect(xpos,ypos,20,10);
}
void drive() {                       // Переміститись
   xpos = xpos + xspeed;
   if (xpos > width) {   xpos = 0;   }
  }
}
```

Об'єкти можна зберігати у **масиві**. Оголошується масив cars об'єктів Car[], у якому резервується пам'ять під 10 таких об'єктів new Car[10].

```
Car[] cars = new Car[10];
```

Для створення цих об'єктів потрібно скористатись циклом:

```
for (int i=0; i<10; i++) {
 cars[i] = new Car(color(255,0,0),i*20,i*20,i+1);
}
```

Для обробки об'єктів теж потрібен цикл:

```
for (int i=0; i<10; i++) {
  cars[i].drive();
  cars[i].display();
}
```

Трансформації

translate (x,y);
Зміни застосовуються до цілої координатної площини, змінюючи точку початку відліку координат.

scale(s);
Якщо s=1, масштаб не змінюється, s<1 - об'єкти зменшуються, s>1 - об'єкти збільшуються.

rotate (angle); rotate (radians(angle));
За умовчанням команда **rotate** здійснює поворот на вказаний кут в радіанах. Для перетворення з числа градусів у радіани використовується команда **radians(angle)**

Трансформація окремих об'єктів

Відбувається через команди **pushMatrix();** та **popMatrix();** між якими записуються команди обробки потрібних об'єктів.

Зображення

Зображення (змінна **img**), яке використовуватиметься в проекті, потрібно оголосити командою, у якій одразу ж вказати значення для цієї змінної (файл із зображенням на диску)

PImage img =loadImage("pic.jpg");

Файл із зображенням **pic.jpg** має розташовуватись у каталозі **data** поточного проекту Processing. Підключити файл до проекту можна через меню **Ескіз - Додати файл** (Sketch - Add File).

Для того, щоб зобразити оголошену картинку на полотні проекту, потрібна команда:

image(img,0,0,640,480);

де **img** - це назва змінної, що містить завантажене зображення, **0,0** - координати її положення на полотні, **640,480** - розміри зображення на полотні (їх можна не вказувати, тоді використовуватимуться справжні розміри зображення).

Функція **img.get(x, y)** дозволяє отримати код кольору пікселя у координатах **x,y**.

Функція **set(x, y, color(255-red(c), 255-green(c), 255-blue(c)))**; дозволяє встановити колір для пікселя **x,y**.

Відео

```
PImage [] images;                  //масив зображень
int currentPosition =0;            // поточний кадр = початковий
void setup() {   size(420,280);
  images = loadImages("movie", ".jpg", 16); // завантаження зображень за маскою
}
void draw(){
 image(images[currentPosition], 0, 0);   //виведення поточного кадру
  currentPosition += 1;                  //зміна кадру на наступний
  if(currentPosition >= images.length)   //якщо кінець масиву кадрів
  {    currentPosition = 0;  }           /то перейти на початковий кадр
}
```

Текст

У службову область:

println("printing some text to the message window! ");println(mouseX);

У вікно проекту:

text("Mmmmm... Strings... " ,10,100);

Текст, що виводиться у вікно проекту може використовувати різні шрифти. Для створення шрифту існує меню **Tool-Create Font** (Інструменти - Створити шрифт):

```
PFont myFont;                    //змінна типу шрифт
void setup (){
  size(500,500);
  myFont = createFont("Georgia", 32); //створення або підключення стандартного шрифта
  textFont(myFont,36);                //визначення розміру шрифта для виведення
  text( "Mmmmm... Strings... " ,10,100); //виведення тексту в координатах
}
```

Рядкова змінна – тип String

String message = "Однакові інтервали між літерами";

message.charAt(3); - значення символу на позиції 3 рядка message.

Символ – тип char

char c = message.charAt(3);

Натиснення кнопок

Функція void keyPressed() викликається 1 раз, щоразу як натиснена будь-яка кнопка. При цьому системна змінна key містить значення останньої натисненої кнопки клавіатури. Стандартні значення кнопок змінної key: BACKSPACE, TAB, ENTER, RETURN, ESC, та DELETE.

```
if (keyPressed) {
   if (key == 'b' || key == 'B') { }          // дві риски позначає АБО
```

Для кнопок за межами таблиці кодування ASCII слід використовувати змінну keyCode. Наприклад, це значення стрілок UP, DOWN, LEFT, та RIGHT, а також ALT, CONTROL, і SHIFT. Перед тим, як перевіряти натиснення цих кнопок є зміст перевірити, чи натиснена кнопка за межами таблиці кодування ASCII:

```
if (key == CODED) {
   if (keyCode == UP) { }
}
```

Хмаринка слів

```
PFont myFont;
IntDict  dictionary;                          //тип даних словник з відповідниками цілих чисел
void setup (){
  size(500,500);
  myFont = createFont("Georgia", 32);
  textFont(myFont,36);
  fill(0);
  String cloud;                               //змінна з текстом
  cloud="Text Oksana mama Text Text";
   String [] words = split(cloud, ' ');       //розділяємо текст за пробілами на окремі слова
  dictionary = new IntDict();                 //починаємо заповнення словника
  for (int i=0; i<words.length; i++) {
     if (dictionary.hasKey(words[i])==true) {dictionary.add(words[i], 1);}
     else {dictionary.set(words[i], 1);}      // якщо слово є у словнику, додаємо 1 до його кількості
    }                                         //інакше, додаємо це слово з кількістю 1
  println(dictionary);                        //друкуємо словник
  dictionary.sortValues();                    //сортуємо словник за кількістю
  String[] theKeys = dictionary.keyArray();          //формуємо рядок з ключів словника - слів
  int n=50;                                   //змінна для зручного друку
  for (int i=0; i<dictionary.size(); i++) {   //для всіх слів словника
    textFont(myFont,(i+1)*20);                //встановлюємо шрифт, розмір якого відповідає номеру слова
    text(theKeys[i],n*(i+2),50*(i+2));        //друкуємо слово відповідного розміру в координатах зі зміщенням
  }
}
```

Маятник

```
float r=0;
float t=0.05;
void setup() {
  size(400, 400);
  strokeWeight(6);
 fill(0);
}
void draw(){
    background(255);
    translate(200, 200);   //переміщуємо початок відліку координат у центр полотна
    rotate(r);             //повертаємо полотно на r радіан
    line(0, 0, 0, 100);    //малюємо лінію від початку координат вниз на 100 пікселів
    ellipse(0, 100, 30,30); //малюємо коло радіусом 30 з центром у завершені лінії
    if ((r>0.8) | (r<-0.8)) {t=-t;}     //якщо досягнули межі руху маятника, змінюємо напрям його руху
    r+=t;                               //змінюємо кут повороту
  }
```