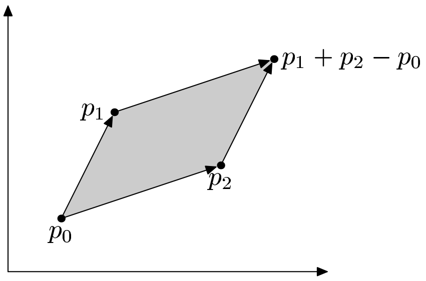
Ćwiczenia 11

Iloczyn wektorowy

Dane trzy punkty $p_0 = (x_0, y_0)$, $p_1 = (x_1, y_1)$ $p_2 = (x_2, y_2)$.

$$(p_1 - p_0) \times (p_2 - p_0) = (x_1 - x_0)(y_2 - y_0) - (x_2 - x_0)(y_1 - y_0)$$



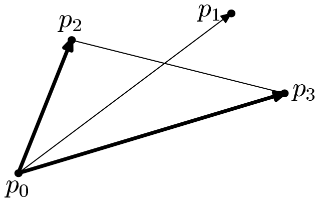
Sprawdzanie, czy dwa odcinki się przecinają

Dane: dwa odcinki p_0—p_1, p_2—p_3
Wynik: 1 — gdy odcinki się przecinają, 0 wpp.

---

1. sprawdzenie czy dwa prostokąty reprezentowane przez odcinki p_0—p_1, i p_1—p_2 mają niepustą część wspólną, jeśli nie to zwróć 0,

2. obliczenie iloczynów wektorowych: $(p_2 - p_0) \times (p_1 - p_0)$ i $(p_3 - p_0) \times (p_1 - p_0)$



3. jeśli ich iloczyn jest liczbą mniejszą bądź równą 0, to oznacza to, że odcinki się przecinają.

Obliczanie pole wielokąta wypukłego

Dane: zbiór kolejnych n wierzchołków wielokąta $p_1, p_2, \ldots, p_n$
Wynik: pole wielokąta

```
area:=0;
p_{n+1}:=p_1;
for i:=1 to n do
    area:=area+(y_{i+1} - y_i)(x_{i+1}+x_i)/2
return abs(area);
```

Sprawdzanie, czy jakakolwiek para odcinków się przecina

Dane: zbiór n odcinków
Wynik: 1 — gdy istnieje para przecinających się odcinków, 0 — wpp

---

Dodatkowe zakładamy o odcinkach, że:

- żadne 3 odcinki nie przecinają się w jednym punkcie,
- nie ma pionowych odcinków.

Algorytm:

T=∅;
posortuj końce odcinków wg współrzędnych x (wg y w przypadku konfliktów)
foreach p_i **do**
  if p - lewy koniec odcinka s
    Insert(T,s)
    if *Above*(T,s) istnieje i przecina s **then return** 1;
    if *Below*(T,s) istnieje i przecina s **then return** 1;
  else /* if p - prawy koniec odcinka s */
    if *Above*(T,s) i *Below*(T,s) istnieją i przecinają się **then**
      return 1
    Remove(T,s);
return 0;

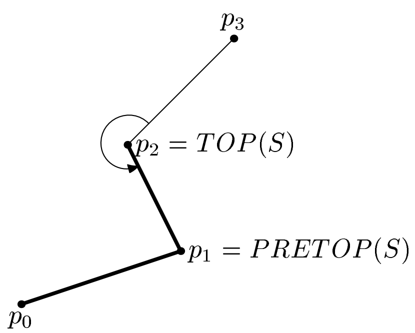
Otoczka wypukła — algorytm Graham'a

Dane: zbiór n punktów
Wynik: wielokąt wypukły o najmniejszym polu, zawierający wszystkie punkty

---

p_0:=punkt o najniższej współrzędnej y, jeśli jest wiele takich punktów,
    wybierz ten o najmniejszej współrzędnej x
posortuj wg współrzędnych biegunowych (dla środka w p_0)
  w przypadku konfliktów jako drugi klucz wybierz odległość od p_0
S=p_0,p_1,p_2 { S - stos }
for i:=3 **to** n **do**
  while kąt p_i,$TOP(S)$,$PRETOP(S)$ $\geq$ 180 **do**
      $POP(S)$
  $PUSH(p_i)$
return S

przed:



po: