logo

szkola

Sprawozdanie

Metody Programowania

Rok akademicki x
Semestr x
Autor: x
Kierunek: x
Grupa x
Laboratorium z
Data zajęć: z
Temat: Geometria obliczeniowa – algorytm Jarvisa
Prowadzący: x

Ocena

Spis treści

1) Wprowadzenie teoretyczne

Geometria obliczeniowa – dział algorytmiki zajmujący się algorytmami i strukturami danych pozwalającymi efektywnie wykonywać działania na obiektach geometrycznych, takich jak zbiory punktów, odcinków, wielokątów, okręgów. Wyniki geometrii obliczeniowej mają istotne znaczenie w wielu dziedzinach informatyki i inżynierii, takich jak grafika komputerowa, robotyka, symulacje komputerowe, bazy danych, projektowanie wspomagane komputerowo.

Algorytm Jarvisa (nazywany również: marsz Jarvisa, owijanie prezentów) - metoda wyznaczania otoczki wypukłej zbioru punktów umieszczonych na płaszczyźnie lub przestrzeni o większej liczbie wymiarów.

Algorytm działa w czasie O(kn),
gdzie: n - całkowita liczba punktów
k - liczba punktów należących do otoczki
W pesymistycznym przypadku złożoność czasowa może wynieść $O(n^2)$

Przy implementacji algorytmów geometrii obliczeniowych należy wykorzystywane są operacje geometryczne takie jak:

a) iloczyn skalarny – służy do obliczania kąta między 2 wektorami, zgodnie ze wzorem:

$$ar\cos\frac{\vec{a}\,o\,\vec{b}}{|\vec{a}|\cdot|\vec{b}|}$$

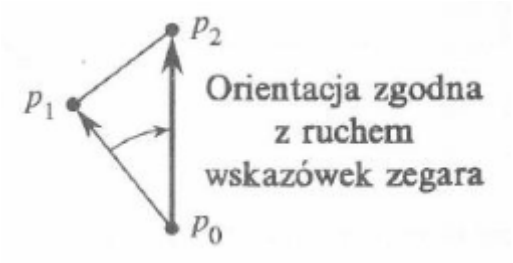
gdzie:

$\vec{a}\,o\,\vec{b}$ – iloczyn skalarny

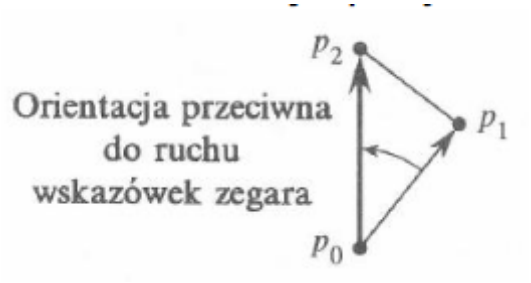
$|\vec{a}|,\ |\vec{b}|$ – długość wektora $|\vec{a}|\ i\ |\vec{b}|$

b) iloczyn wektorowy – umożliwia ustalenie orientacji dwóch wektorów

Jeśli $\vec{a}\,x\,\vec{b} > 0$ to orientacja zgodna z ruchem wskazówek zegara



Jeśli $\vec{a} x \vec{b} < 0$ to orientacja przeciwna do ruchu wskazówek zegara



Jeśli $\vec{a} x \vec{b} = 0$ to wektory są współliniowe

gdzie:

$\vec{a} x \vec{b}$ – iloczyn wektorowy

2) Problemy do rozwiązania

Należy napisać program implementujący algorytm Jarvisa, użytkownik powinien mieć możliwość podania współrzędnych punktów z konsoli lub pliku txt, wynik powinien być wyświetlany oraz (jeśli użytkownik chce) zapisywany do osobnego pliku txt .
Należy sprawdzić poprawność programu oraz podać przykładowe wejścia/wyjścia.
W celu czytelniejszej reprezentacji punktu w kodzie można stworzyć klasę Point mającą zmienne składowe x i y.
Należy również napisać funkcje pomocnicze:

a. obliczającą kąt między wektorami
b. wyznaczającą ich orientację.
c. generator losowych punktów

3) Program implementujący algorytm Jarvisa

a) Pseudokod

wejście: punkty (o dwóch współrzędnych), ilość punktów

wyjście: punkty należące do otoczki, wypisywane od punktu o minimalnej współrzędnej y idąc w kierunku przeciwnym do ruchu wskazówek zegara

Opis działania:

1. Wyznacz punkt P[0] na otoczce wypukłej o najmniejszej współrzędnej y (jeśli jest więcej niż jeden, wybierany jest ten o najmniejszej współrzędnej x)
2. Wyznacz punkt Q[0] na otoczce wypukłej o największej współrzędnej y (jeśli jest więcej niż jeden, wybierany jest ten o największej współrzędnej x),
3. Wyznacz prawy łańcuch otoczki:
    1. i = 0
    2. Powtarzaj:
        a. S = P[i]
        b. Wyznacz punkt N dla którego kąt między wektorem $\overrightarrow{SN}$ a wektorem [1,0] jest najmniejszy, N należy do otoczki,
        c. P[i+1] = N
        d. Jeśli N = Q[0] koniec iterowania
        e. i++:
4. Wyznacz lewy łańcuch otoczki:
    1. i = 0
    2. Powtarzaj:
        a. S = Q[i]
        b. Wyznacz punkt N dla którego kąt między wektorem $\overrightarrow{SN}$ a wektorem [-1,0] jest najmniejszy, N należy do otoczki,
        c. Q[i+1] = N
        d. Jeśli N = P[0] koniec iterowania
        e. i++
5. Otoczkę wypukłą określają punkty P i Q (P zawiera punkty należące do prawego łańcucha otoczki, Q do lewego)
6. Połącz tablice P i Q, a następnie wyprowadź

b) C++

Program został napisany w języku C++, obok linijek z kodem (lub nad nimi) po sekwencji „//” znajdują się komentarze do danego fragmentu.

Program został skompilowany oraz uruchomiony na sprzęcie/oprogramowaniu:

procesor CPU: Intel i5-4210H

system operacyjny: Windows 10 Home wersja: 20H2

kompilator: g++ wersja 9.2.0

Kod źródłowy znajduje się w pliku *main.cpp*

c) Przykładowe wejścia / wyjścia z programu

Wejście:

Punkt 0: 583 973	Punkt 28: 508 707	Punkt 56: 940 734	Punkt 84: 46 835
Punkt 1: 377 365	Punkt 29: 478 624	Punkt 57: 842 936	Punkt 85: 225 709
Punkt 2: 565 789	Punkt 30: 747 53	Punkt 58: 956 529	Punkt 86: 102 512
Punkt 3: 396 345	Punkt 31: 499 284	Punkt 59: 487 81	Punkt 87: 122 13
Punkt 4: 212 993	Punkt 32: 875 480	Punkt 60: 724 57	Punkt 88: 429 375
Punkt 5: 624 921	Punkt 33: 176 918	Punkt 61: 221 320	Punkt 89: 126 441
Punkt 6: 583 499	Punkt 34: 483 480	Punkt 62: 859 986	Punkt 90: 867 930
Punkt 7: 352 743	Punkt 35: 603 712	Punkt 63: 668 541	Punkt 91: 895 558
Punkt 8: 300 561	Punkt 36: 346 380	Punkt 64: 544 101	Punkt 92: 694 40
Punkt 9: 700 443	Punkt 37: 864 287	Punkt 65: 355 242	Punkt 93: 128 852
Punkt 10: 442 414	Punkt 38: 887 331	Punkt 66: 207 945	Punkt 94: 930 349
Punkt 11: 852 609	Punkt 39: 179 522	Punkt 67: 539 821	Punkt 95: 565 581
Punkt 12: 242 731	Punkt 40: 682 348	Punkt 68: 306 193	Punkt 96: 728 963
Punkt 13: 731 214	Punkt 41: 1 687	Punkt 69: 900 672	Punkt 97: 226 474
Punkt 14: 609 102	Punkt 42: 383 746	Punkt 70: 980 711	Punkt 98: 474 814
Punkt 15: 562 659	Punkt 43: 160 227	Punkt 71: 367 37	Punkt 99: 771 666
Punkt 16: 757 755	Punkt 44: 589 504	Punkt 72: 498 697	
Punkt 17: 810 253	Punkt 45: 436 777	Punkt 73: 955 574	
Punkt 18: 205 435	Punkt 46: 82 377	Punkt 74: 971 252	
Punkt 19: 727 500	Punkt 47: 971 636	Punkt 75: 916 116	
Punkt 20: 62 947	Punkt 48: 177 923	Punkt 76: 330 383	
Punkt 21: 561 170	Punkt 49: 179 208	Punkt 77: 281 638	
Punkt 22: 603 70	Punkt 50: 438 621	Punkt 78: 889 721	
Punkt 23: 172 255	Punkt 51: 434 22	Punkt 79: 493 245	
Punkt 24: 221 717	Punkt 52: 369 343	Punkt 80: 496 645	
Punkt 25: 250 491	Punkt 53: 350 782	Punkt 81: 304 646	
Punkt 26: 908 350	Punkt 54: 786 880	Punkt 82: 414 772	
Punkt 27: 543 987	Punkt 55: 607 810	Punkt 83: 990 159	

Wyjście (współrzędne punktów na otoczce)

122 13
434 22
694 40
747 53
916 116
990 159
980 711
859 986
212 993
62 947
1 687

Czas wykonania: 18.975 s

Wejście:

Punkt 0: 598 835	Punkt 28: 249 91	Punkt 56: 474 286	Punkt 84: 120 188
Punkt 1: 523 526	Punkt 29: 778 808	Punkt 57: 963 921	Punkt 85: 645 923
Punkt 2: 665 518	Punkt 30: 122 82	Punkt 58: 548 206	Punkt 86: 486 815
Punkt 3: 466 457	Punkt 31: 240 950	Punkt 59: 396 676	Punkt 87: 675 155
Punkt 4: 15 724	Punkt 32: 54 302	Punkt 60: 751 931	Punkt 88: 670 74
Punkt 5: 670 330	Punkt 33: 258 459	Punkt 61: 14 527	Punkt 89: 404 719
Punkt 6: 14 967	Punkt 34: 367 653	Punkt 62: 673 798	Punkt 90: 6 19
Punkt 7: 893 209	Punkt 35: 63 890	Punkt 63: 697 931	Punkt 91: 898 281
Punkt 8: 162 428	Punkt 36: 18 623	Punkt 64: 277 410	Punkt 92: 660 302
Punkt 9: 141 709	Punkt 37: 890 454	Punkt 65: 394 524	Punkt 93: 634 467
Punkt 10: 688 896	Punkt 38: 342 302	Punkt 66: 272 292	Punkt 94: 899 568
Punkt 11: 657 239	Punkt 39: 868 115	Punkt 67: 338 115	Punkt 95: 216 504
Punkt 12: 77 486	Punkt 40: 544 324	Punkt 68: 187 388	Punkt 96: 75 969
Punkt 13: 137 631	Punkt 41: 120 621	Punkt 69: 32 270	Punkt 97: 347 170
Punkt 14: 115 776	Punkt 42: 941 916	Punkt 70: 314 171	Punkt 98: 500 232
Punkt 15: 137 366	Punkt 43: 675 336	Punkt 71: 81 166	Punkt 99: 514 604
Punkt 16: 220 730	Punkt 44: 391 554	Punkt 72: 156 855	
Punkt 17: 757 292	Punkt 45: 932 328	Punkt 73: 981 913	
Punkt 18: 418 365	Punkt 46: 836 132	Punkt 74: 573 526	
Punkt 19: 163 386	Punkt 47: 592 810	Punkt 75: 651 306	
Punkt 20: 654 426	Punkt 48: 915 757	Punkt 76: 856 224	
Punkt 21: 602 186	Punkt 49: 897 67	Punkt 77: 824 574	
Punkt 22: 935 748	Punkt 50: 817 504	Punkt 78: 553 47	
Punkt 23: 690 118	Punkt 51: 767 287	Punkt 79: 220 279	
Punkt 24: 565 36	Punkt 52: 643 434	Punkt 80: 426 283	
Punkt 25: 228 958	Punkt 53: 831 436	Punkt 81: 941 534	
Punkt 26: 262 767	Punkt 54: 285 443	Punkt 82: 740 881	
Punkt 27: 962 911	Punkt 55: 316 729	Punkt 83: 536 821	

wyjście

6 19
565 36
897 67
932 328
981 913
963 921
75 969
14 967

Czas wykonania: 40.34 s

d) Testowanie poprawności

W celu sprawdzenia poprawności programu najpierw narysuje punkty na płaszczyźnie 2d w programie *geogebra*, wybiorę punkty otoczki, następnie uruchomię program i porównam wyniki.

Test 1

Dane wejściowe (wygenerowane losowo)

Punkt 0: 447 278

Punkt 1: 203 30

Punkt 2: 806 918

Punkt 3: 180 442

Punkt 4: 905 348

Punkt 5: 895 821

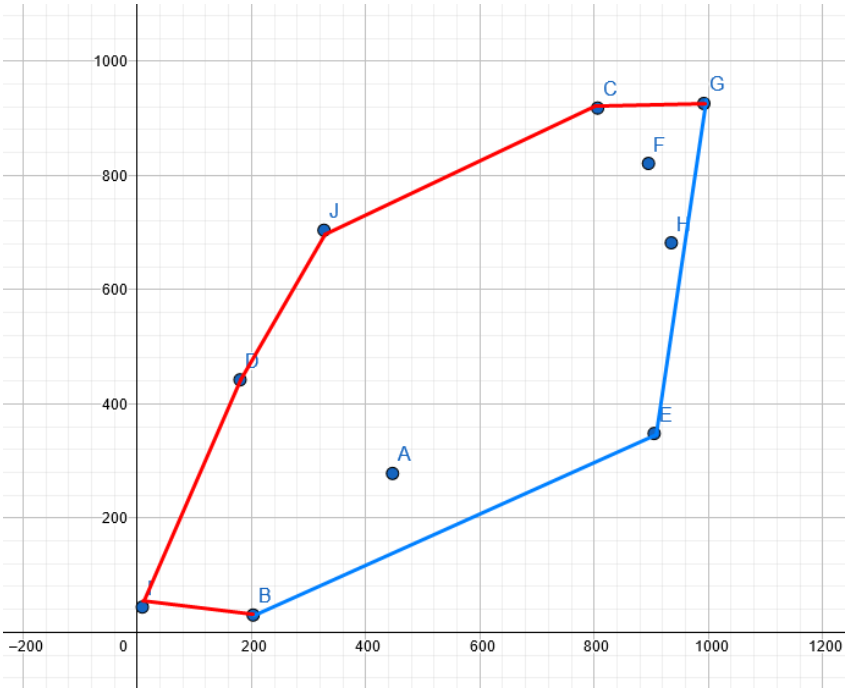
Punkt 6: 992 926

Punkt 7: 955 682

Punkt 8: 9 44

Punkt 9: 327 704

Przedstawienie punktów graficznie oraz samodzielne wyznaczenie otoczki



Wyjście programu

203 30

905 348

992 926

806 918

327 704

180 442

9 44

Ocena poprawności: Działa poprawnie

Test 2

Dane wejściowe

Punkt 0: 0 3

Punkt 1: 2 2

Punkt 2: 1 1

Punkt 3: 2 1

Punkt 4: 3 0

Punkt 5: 0 0

Punkt 6: 3 3

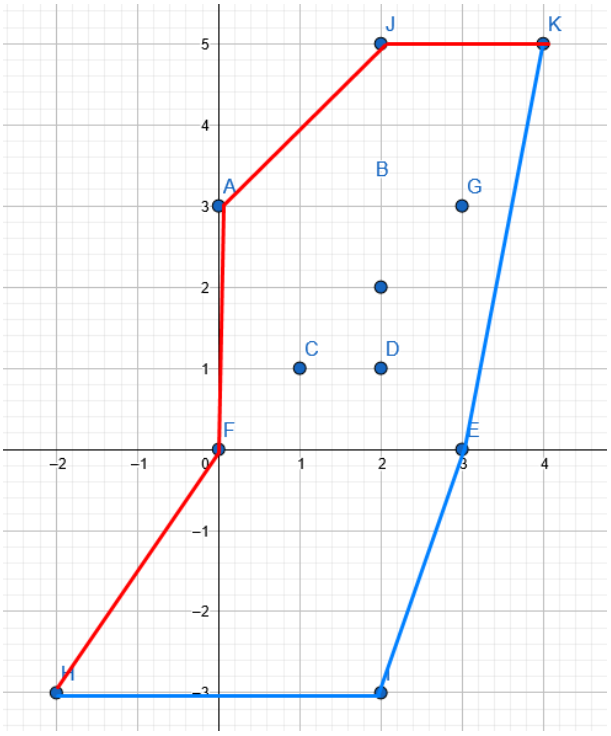
Punkt 7: -2 -3

Punkt 8: 2 -3

Punkt 9: 2 5

Punkt 10: 4 5

Przedstawienie punktów graficznie oraz samodzielne wyznaczenie otoczki



Wyjście programu

-2 -3

2 -3

3 0

4 5

2 5

0 3

Ocena poprawności: Działa poprawnie, jeśli jest kilka punktów o najmniejszym y, to wybierany jest ten z mniejszym x.

Test 3

Dane wejściowe (wygenerowane losowo)

4.7 57

362 201

707 -99

4.88 -748

581 865

-16 191

347 394

7 654

99 22

-87.2 102

78 255

5.87 -14.8

302 123

-429 -81

458 469

26 441

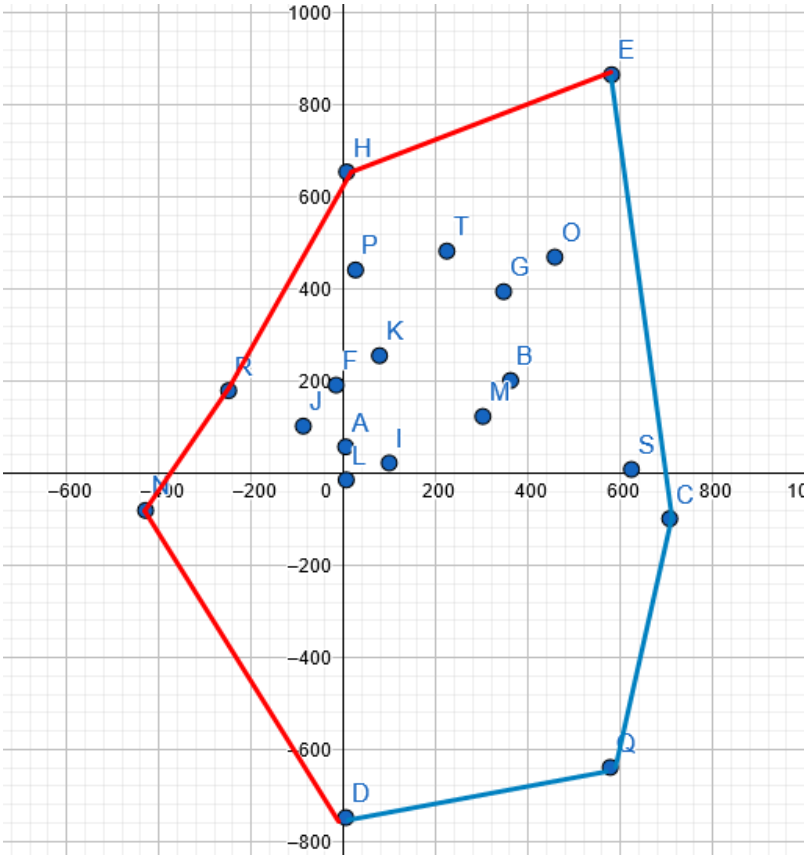
578 -639

-249 179

624 8

224 482

Przedstawienie punktów graficznie oraz samodzielne wyznaczenie otoczki



Wyjście programu

Ocena poprawności: Działa poprawnie, program obsługuje również liczby przecinkowe oraz ujemne

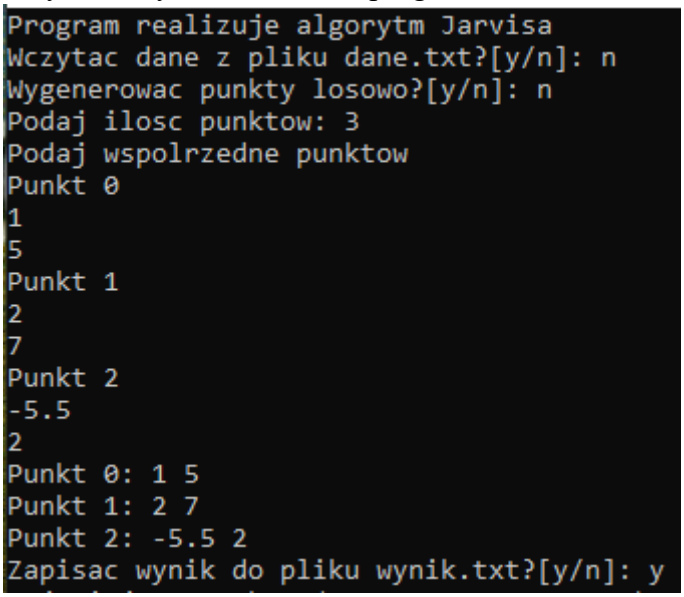
Wniosek:

Program działa poprawnie

4) Obsługa programu

Po uruchomieniu programu bez argumentów wyświetla się informacja o programie. Następnie użytkownik zostaje zapytany czy wczytać dane z pliku, jeśli wprowadzony zostanie znak odmowy ‘n’ wyświetlone zostanie kolejne pytanie czy użytkownik chce, by punkty zostały wygenerowane losowo, jeśli ‘n’ to użytkownik podaje ilość punktów, a następnie wpisuje współrzędne punktów z klawiatury.
Później użytkownik zostaje zapytany czy chce zapisać wyniki do pliku.
Następuje wykonanie programu, zostaje wyświetlony (ewentualnie zapisany do pliku) wynik

Przykładowy zrzut ekranu z programu



5) Wnioski

Program przetestowano i wyciągnięto wniosek, że działa poprawnie. Im więcej punktów tym dłuższy czas wykonania programu.

Program umożliwia wprowadzenie punktów o współrzędnych typu *double*. Otoczka tworzy się prawidłowo.
Najpierw wyznaczany jest punkt o najmniejszym y, następnie tworzony jest prawy łańcuch aż do punktu o największym y, następnie tworzony jest lewy łańcuch aż do punktu początkowego, łańcuchy są łączone i otrzymujemy punkty na otoczce wypukłej podanego zbioru punktów.

6) Literatura

[1] https://pl.wikipedia.org/wiki/Algorytm_Jarvisa (dostęp: 29.05.2021)

[2] https://www.geeksforgeeks.org/convex-hull-set-1-jarviss-algorithm-or-wrapping/ (dostęp: 29.05.2021)

[3] https://www.geeksforgeeks.org/orientation-3-ordered-points/ (dostęp: 29.05.2021)

[4] https://pl.wikipedia.org/wiki/Iloczyn_wektorowy (dostęp: 29.05.2021)

[5] https://matematykaszkolna.pl/strona/1630.html (dostęp: 29.05.2021)

[6] https://pl.wikipedia.org/wiki/Geometria_obliczeniowa (dostęp: 29.05.2021)