```
/*****************************************************************
   ***************
*
    *
*   Algorytm Grahama konstrukcji wypukłej otoczki                 *
*   Plik pobrany ze strony:          www.algorytm.org
    *
*   Opracowal:              Michal Knasiecki      *
*
    *
*****************************************************************
   ***************/

/*Uwagi:

1. W poniższej implementacji posłużono się biblioteką STL, która
   zawiera szablon wektora oraz stosu.
   Jeżeli twój kompilator nie posiada tej biblioteki możesz
      zastąpić
   wektor tablicą oraz wykorzystać własnąimplementacjęstosu.

2. Dane wejściowe muszą być w postaci uporządkowanej (patrz strona
   WWW).

3. Wynik jest wypisywany w porządku zgodnym z kierunkiem ruchu
   wskazówek zegara.
*/

#include <stdio.h>
#include <stdlib.h>
#include <conio.h>

#include <stack>
#include <vector>

using namespace std;
typedef struct
    {
   int x;
   int y;
   } Tpunkt;

//Funkcja oblicza wyznacznik macierzy 3x3 (metodą Sarrusa)
int det(Tpunkt p1, Tpunkt p2, Tpunkt p3){
return p1.x*p2.y + p2.x*p3.y + p3.x*p1.y - p3.x*p2.y - p1.x*p3.y -
   p2.x*p1.y;
}
```

```
//Funkcja sprawdza, czy przechodząc do punktu p3 skręcamy w prawo
   (1), czy w lewo (0)
int skret_w_prawo(stack<Tpunkt,vector<Tpunkt> > S,Tpunkt p3){
Tpunkt p2;
Tpunkt p1;

//Pobieramy ze szczytu stosu punkt p2 oraz punkt p1, bedacy tuż
   pod szczytem
p2=S.top();
S.pop();
p1=S.top();
S.push(p2);

//Punkt p3 leży po lewej stronie wektora p1->p2
if (det(p1,p2,p3)>0)
    return 0;

//Punkt p3 leży po prawej stronie wektora p1->p2
if (det(p1,p2,p3)<0)
    return 1;
}

void main(void){
//Lista wejściowa (uporządkowana: czyt. na stronie)
vector<Tpunkt> Q;

//Stos punktów, na końcu zawiera wynik
stack<Tpunkt,vector<Tpunkt> > S;

Tpunkt punkt;
int liczba_punktow;

//Przykładowa instancja danych (zob. przyklad.gif)

liczba_punktow=10;
// punkt a
punkt.x=1;
punkt.y=1;
Q.push_back(punkt);
// punkt b
punkt.x=6;
punkt.y=2;
Q.push_back(punkt);
// punkt c
punkt.x=3;
punkt.y=2;
Q.push_back(punkt);
```

```
// punkt d
punkt.x=4;
punkt.y=3;
Q.push_back(punkt);
// punkt e
punkt.x=5;
punkt.y=6;
Q.push_back(punkt);
// punkt f
punkt.x=3;
punkt.y=5;
Q.push_back(punkt);
// punkt g
punkt.x=4;
punkt.y=8;
Q.push_back(punkt);
// punkt h
punkt.x=2;
punkt.y=6;
Q.push_back(punkt);
// punkt i
punkt.x=1;
punkt.y=7;
Q.push_back(punkt);
// punkt j
punkt.x=0;
punkt.y=4;
Q.push_back(punkt);

//Początek algorytmu

//Umieszczamy 3 pierwsze punkty na stosie
S.push(Q[0]);
S.push(Q[1]);
S.push(Q[2]);

for (int i=3;i<liczba_punktow;i++){
    while (skret_w_prawo(S,Q[i])==1)
    S.pop();
   S.push(Q[i]);
}


printf("Punkty tworzace wypukla otoczke: \n");
while (!(S.empty()))
{
punkt=S.top();
S.pop();
```

```
printf("(%d,%d) ",punkt.x,punkt.y);
}
printf("\nDowolny klawisz...");
getch();
return;
}
```