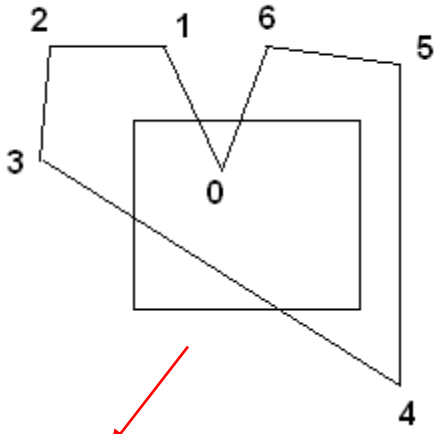
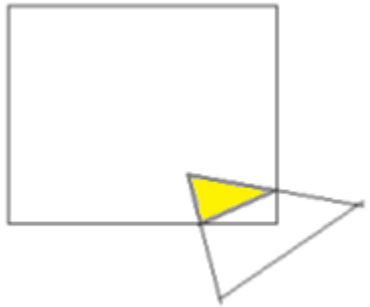
Decuparea poligoanelor

Prof. univ. dr. ing. Florica Moldoveanu

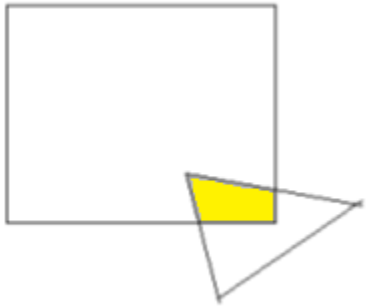
Curs *Elemente de Grafică pe Calculator* – UPB, Automatică şi Calculatoare
2020-2021

Decuparea poligoanelor 2D (1)

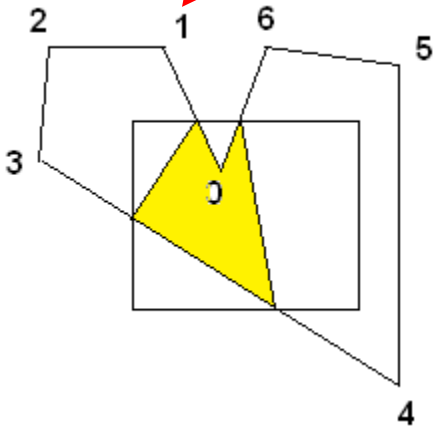


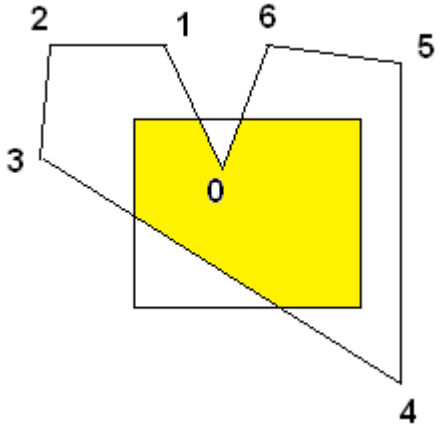


Incorect



Corect

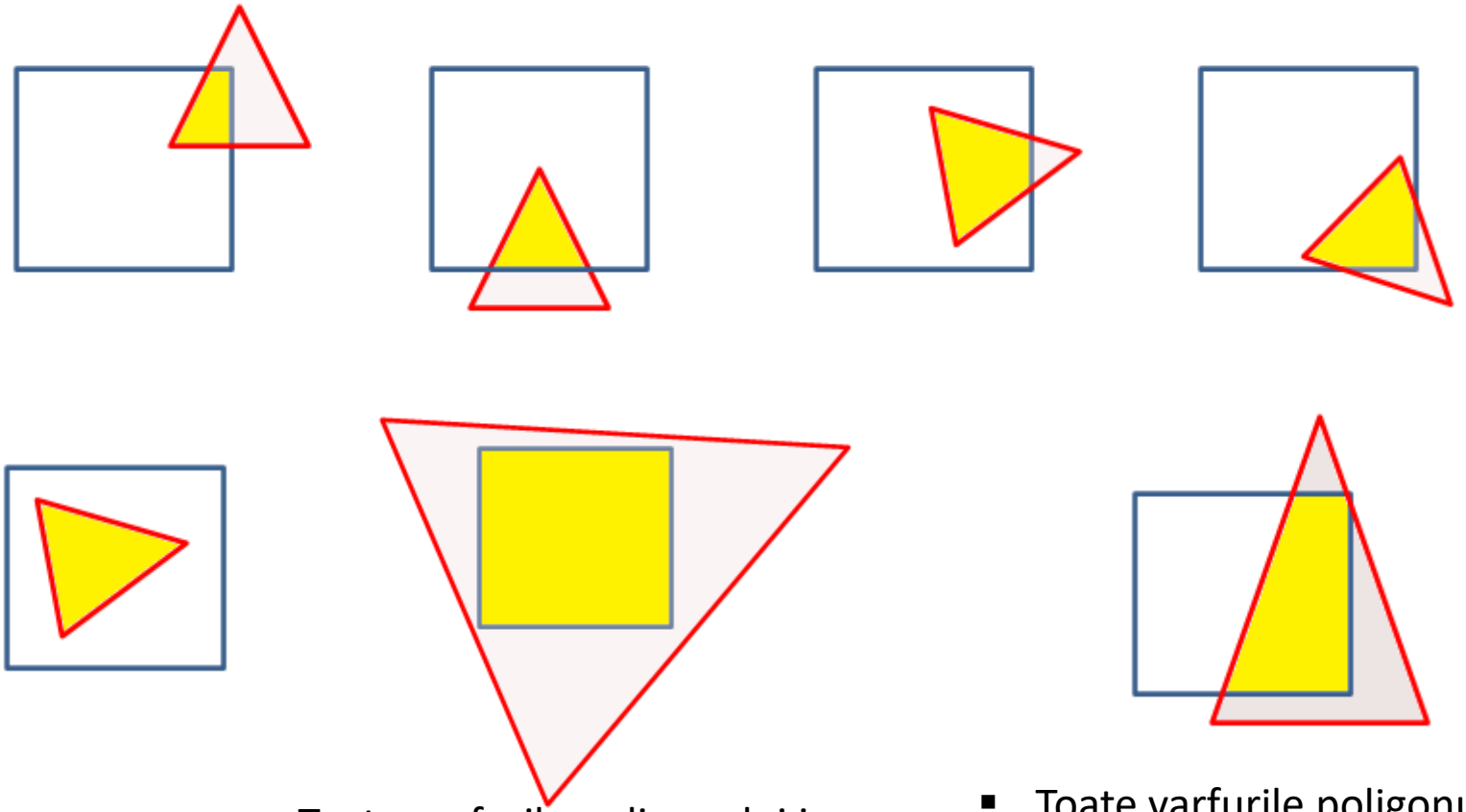




Rezultatul obtinut daca se decupeaza pe rand laturile poligonului folosind algoritmul de decupare vectori.

Rezultatul corect.

Decuparea poligoanelor 2D (2)

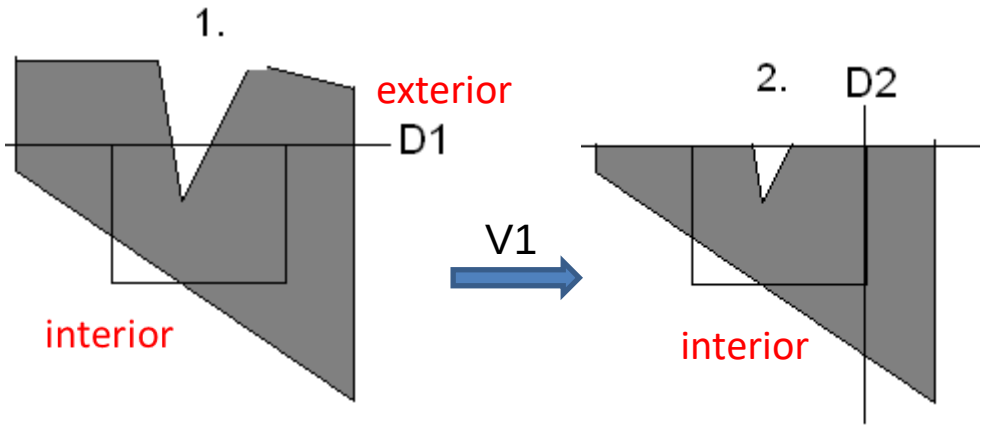


- Toate varfurile poligonului in exteriorul drept. de decupare
- Drept. de decupare inclus in suprafata poligonului

- Toate varfurile poligonului in exteriorul drept. de decupare
- Suprafata drept. de decupare este intersectata de suprafata poligonului

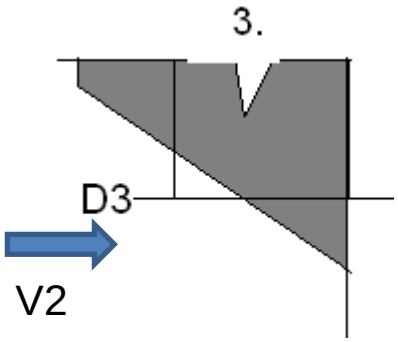
Algoritmul Sutherland-Hodgman(1)

Decuparea poligoanelor convexe față de o zona dreptunghiulară

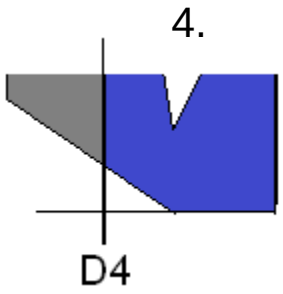


2. Se decupeaza poligonul cu varfurile din lista V1 fata de dreapta D2
   - Se creaza lista varfurilor poligonului rezultat ,V2

1. Se decupeaza laturile poligonului fata de dreapta pe care se afla o latura a zonei de decupare, D1
   - Se creaza lista varfurilor poligonului rezultat (partea din poligonul initial cuprinsa in semiplanul interior), V1



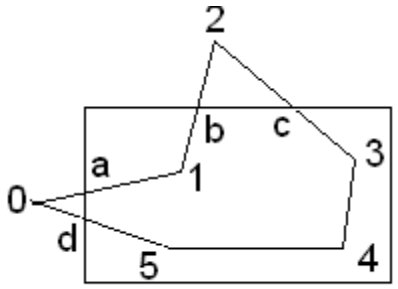
3. Se decupeaza poligonul cu varfurile din lista V2 fata de dreapta D3
   - Se creaza lista varfurilor poligonului rezultat, V3



4. Se decupeaza poligonul cu varfurile din lista V3 fata de dreapta D4
   - Se creaza lista varfurilor poligonului rezultat ,V4

Algoritmul Sutherland-Hodgman(2)

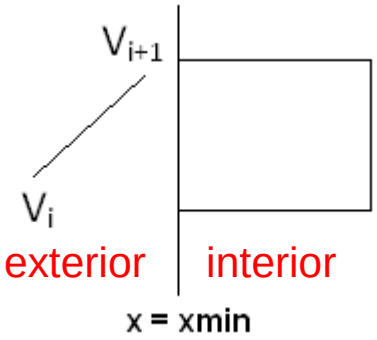
Calculul varfurilor poligonului rezultat intr-o etapa a decuparii



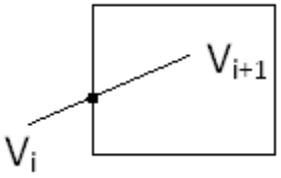
0-1: a, 1 --> a, 1
1-2: 1,b --> a, 1, 1, b
2-3: c,3 --> a, 1, 1, b, c, 3
3-4: 3,4 --> a,1, 1, b, c, 3, 3, 4
4-5: 4,5 --> a,1, 1, b, c, 3, 3, 4, 4, 5
5-0: 5,d --> a,1, 1, b, c, 3, 3, 4, 4, 5, 5, d

Lista corecta: a, 1, b, c, 3, 4, 5, d

Convenție:
- Fiecare latura a poligonului se considera un interval: [V_i -> V_{i+1})
- Numai varful V_i apartine laturii

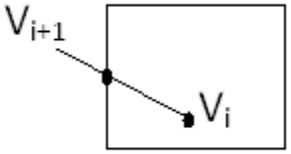


0 varfuri



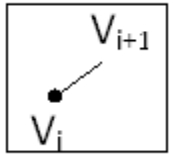
Latura intra in semiplanul interior

un varf



Latura iese din semiplanul interior

2 varfuri

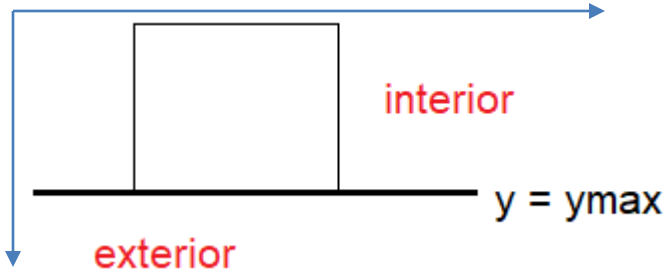


un varf

Algoritmul Sutherland-Hodgman(3)

int DecupLatura(P2D v1, P2D v2, P2d * v1d, P2D *v2d, int latura)

// decupeaza latura v1→v2, fata de dreapta pe care se afla o latura a zonei de decupare




```
{ // functia intoarce numarul de varfuri rezultate din decupare
  switch(latura)
  { case 1: // dreapta y = ymax
      if(v1.y > ymax && v2.y > ymax) return 0; // latura in semiplanul exterior al zonei de decupare
      if(v1.y <= ymax && v2.y <= ymax) { *v1d = v1; return1;} // latura in semiplanul interior
      if(v1.y <= ymax && v2.y > ymax) // latura iese din semiplanul interior
          { * v1d = v1; x_intersectie = …; v2d->x = x_intersectie; v2d->y = ymax; return 2;}
      else// v1.y > ymax && v2.y <= ymax  // latura intra in semiplanul interior
         { x_intersectie = ….; v1d->x = x_intersectie; v1d->y = ymax; return 1;}
    case 2:  ……..
  }
}
```

Algoritmul Sutherland-Hodgman(4)

```
void DecupLaturi(int nv, P2D *vrf, int *nd, P2D *vrfd, int latura)
{ // decupeaza toate laturile poligonului fata de dreapta pe care se afla o latura a zonei de
     decupare
 // vrf: lista varfurilor poligonului de decupat
//  nv: numarul de varfuri ale poligonului de decupat
 // vrfd: lista varfurilor poligonului rezultat din decupare
//  nd: numarul de varfuri ale poligonului rezultat din decupare


   *nd = 0;

   for(int i =0; i< nv; i++)

      *nd+=  DecupLatura(vrf[i], vrf[i+1], &vrfd[*nd], &vrfd[*nd+1], latura);

  *nd+= DecupLatura(vrf[nv-1], vrf[0], &vrfd[*nd], &vrfd[*nd + 1], latura);

 }
```

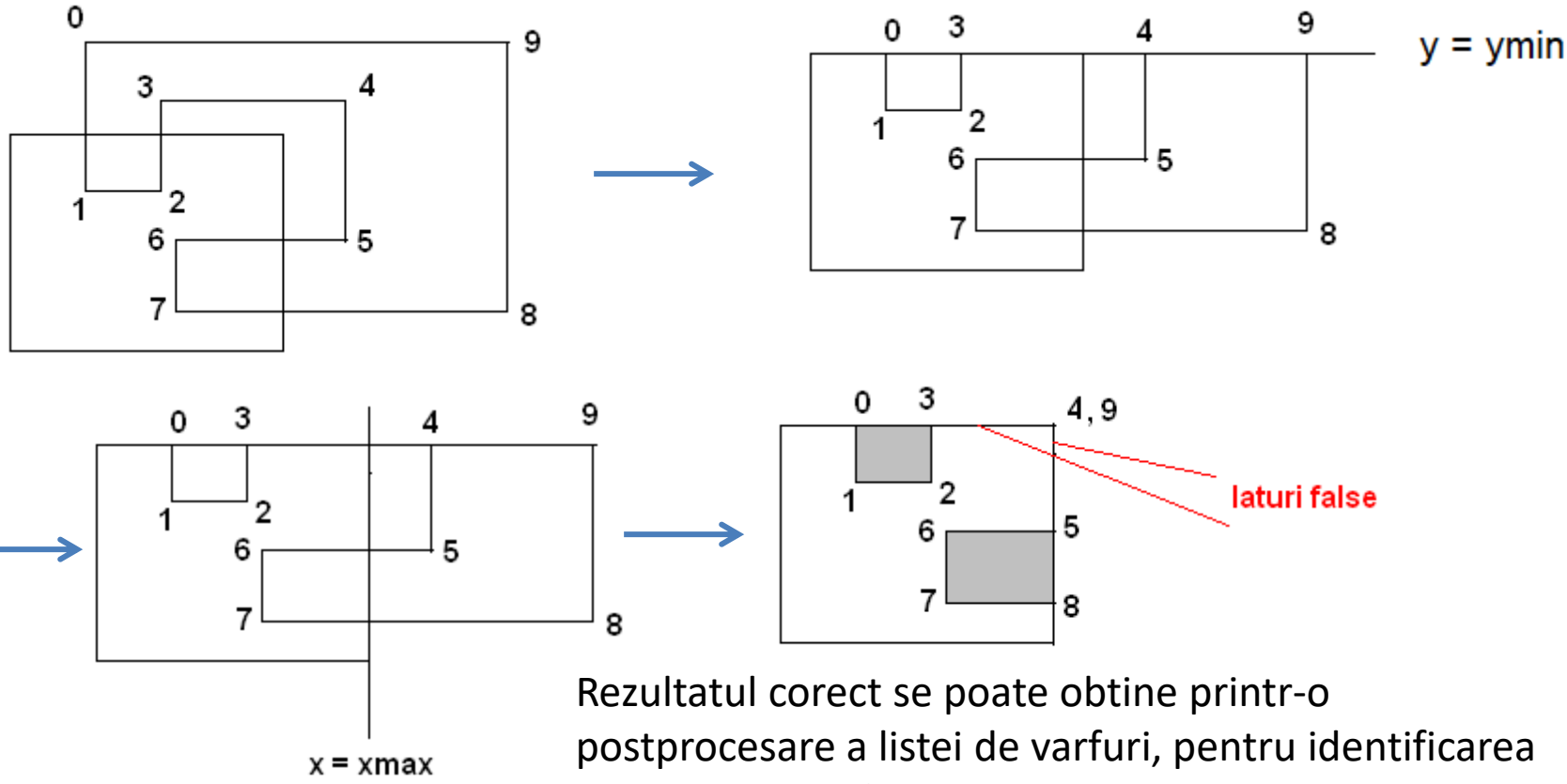
Algoritmul Sutherland-Hodgman(5)

```
void DecupPoligon(int nv, P2D * vrf, int *nd, P2D **vrfd)
{ // vrf: vectorul varfurilor poligonului de decupat
 // vrfd: vectorul varfurilor poligonului rezultat din decupare
  if (poligonInterior(nv, vrf, xmin, xmax, ymin, ymax)) // toate varfurile poligonului de decupat sunt in
                                                        //   interiorul dreptunghiului de decupare
      {*nd = nv; *vrfd = new P2D[nv]; memcpy(*vrfd, vrf, nv*sizeof(P2D));
       return; }
  P2D * v1 = new P2D[2*nv]; P2D * v2 = new P2D[2*nv]; // se folosesc 2 liste de varfuri
  memcpy(v1, vrf, nv*sizeof(P2D));
  for(int latura =1 ; latura <= 4; latura+ =2) // zona de decupare un dreptunghi
  { DecupLaturi(nv, v1, nd, v2, latura);
    DecupLaturi(*nd, v2, &nv, v1, latura+1);
  }
*nd = nv;
*vrfd = new P2D[nv];
  memcpy(*vrfd, v1, nv*sizeof(P2D));
  delete v1; delete v2;
}
```

In cazul unei zone de decupare poligon oarecare, decuparea se efectueaza faţă de fiecare latura a poligonului.

Algoritmul Sutherland-Hodgman(6)

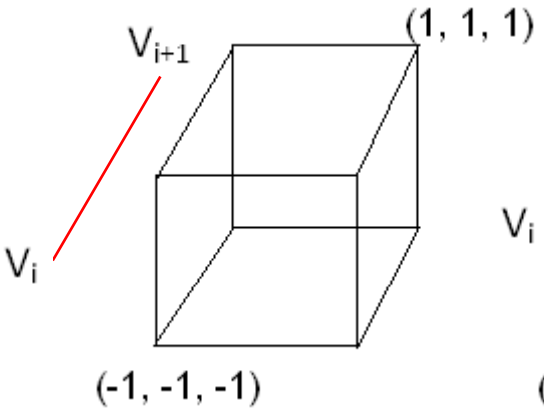
Limitarea algoritmului Sutherland-Hodgman: la decuparea poligoanelor concave, atunci cand din decupare pot rezulta mai multe poligoane. Algoritmul produce o singura lista de varfuri: un singur poligon, cu posibile laturi false.



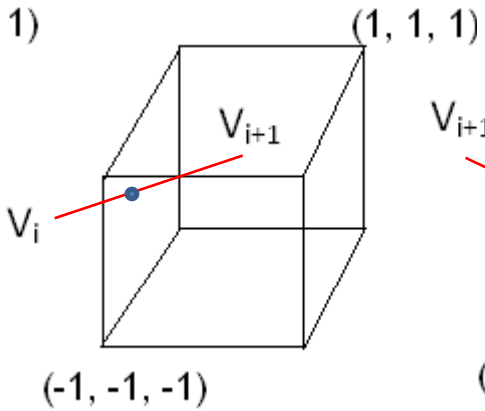
Rezultatul corect se poate obtine printr-o postprocesare a listei de varfuri, pentru identificarea ciclurilor de varfuri.

Decuparea poligoanelor 3D prin algoritmul Sutherland-Hodgman(1)

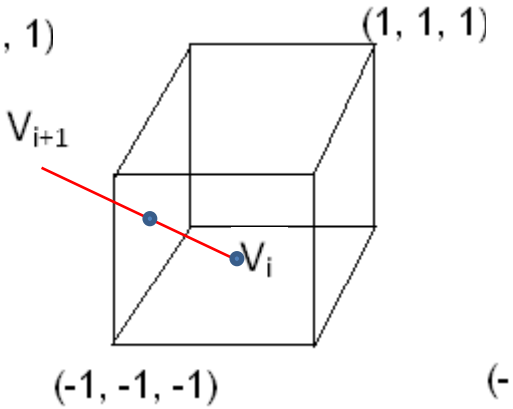
- Decuparea unui poligon 3D se efectueaza la frontiera volumului canonic.
- Se intersecteaza poligonul, pe rand, **cu cele 6 planuri** care delimiteaza volumul.
- Fiecare plan de intersectie imparte spatiul in 2 semispatii:
  - semispatiul interior – care conține volumul
  - semispatiul exterior
- Din fiecare intersectie pot rezulta noi varfuri. Exemplu: intersectia cu planul x=-1



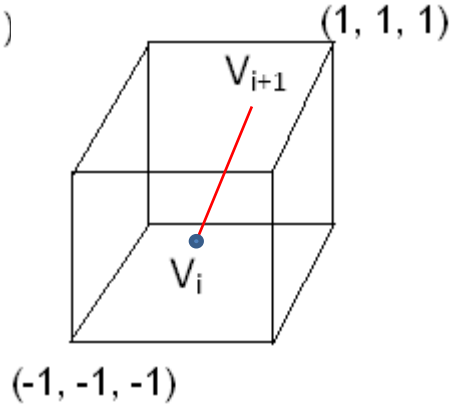
0 varfuri



1 varf: intersectia vectorului cu planul x=-1



2 varfuri: Vi si intersectia vectorului cu planul x=-1



1 varf: varful Vi

Decuparea poligoanelor 3D prin algoritmul Sutherland-Hodgman(2)

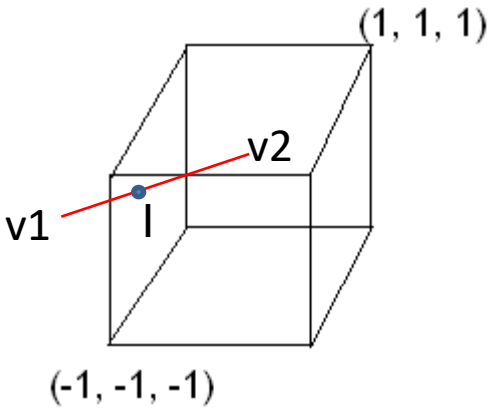
Intersectia unei laturi V1-V2 cu planul x=-1

Ec. parametrice ale laturii

$x = x1 + t(x2 - x1)$

$y = y1 + t(y2 - y1)$

$z = z1 + t(z2 - z1)$



Intersectia cu planul x=-1

$-1 = x1 + t(x2-x1)$ → tinters = $-(x1+1)/(x2-x1)$

$xi = -1$

$yi = y1 + tinters * (y2-y1)$

$zi = z1 + tinters * (z2-z1)$

Decuparea poligoanelor 3D prin algoritmul Sutherland-Hodgman(3)

```
int DecupLatura(P3D v1, P3D v2, P3d * v1d, P3D *v2d, int plan)
// decupeaza latura v1->v2 fata de planul in care se afla o față a volumului de decupare
{ // functia intoarce numarul de varfuri rezultate din decupare
  float tint;
  switch(plan)
  { case 1: // planul x = -1
    if(v1.x < -1 && v2.x < -1) return 0; // latura in semispatiul exterior

    if(v1.x >= -1 && v2.x >= -1 ) { *v1d = v1; return1;} // latura in semispatiul interior

    if(v1.x <-1 && v2.x >= -1) // latura intra in semispatiul interior
    {tint = -(v1.x+1)/(v2.x-v1.x);  v1d->x = -1; v1d->y = v1.y + tint*(v2.y-v1.y); v1d->z= v1.z + tint*(v2.z-v1.z);
    return 1;}
    else // v1.x >= -1 && v2.x  < -1) // latura iese din semispatiul interior
    { * v1d = v1; tint = -(v1.x+1)/(v2.x-v1.x);  v2d->x = -1; v2d->y = v1.y + tint*(v2.y-v1.y);
        v2d->z= v1.z + tint*(v2.z-v1.z) ; return 2;}
   case 2: …….
…… } …….}
```

Decuparea poligoanelor 3D prin algoritmul Sutherland-Hodgman(4)

void DecupLaturi(int nv, P3D *vrf, int *nd, P3D *vrfd, int plan)

{ // decupeaza toate laturile poligonului fata de planul in care se afla o față a volumului de decupare

// vrf: lista varfurilor poligonului de decupat

// nv: numarul de varfuri ale poligonului de decupat

// vrfd: lista varfurilor poligonului rezultat din decupare

// nd: numarul de varfuri ale poligonului rezultat din decupare

```
  *nd = 0;

  for(int i =0; i< nv; i++)

     *nd+=  DecupLatura(vrf[i], vrf[i+1], &vrfd[*nd], &vrfd[*nd+1], plan);

 *nd+= DecupLatura(vrf[nv-1], vrf[0], &vrfd[*nd], &vrfd[*nd + 1], plan);

}
```

Decuparea poligoanelor 3D prin algoritmul Sutherland-Hodgman(5)

```
void DecupPoligon(int nv, P3D * vrf, int *nd, P3D **vrfd)
{ // vrf: lista varfurilor poligonului de decupat
 // vrfd: lista varfurilor poligonului rezultat din decupare
// daca toate varfurile poligonului de decupat sunt in interiorul volumului de decupare
   if (poligonInterior(nv, vrf, xmin, xmax, ymin, ymax, zmin, zmax))
      {*nd = nv; *vrfd = new P2D[nv]; memcpy(*vrfd, vrf, nv*sizeof(P2D)); return; }
   P3D * v1 = new P3D[2*nv]; P3D * v2 = new P3D[2*nv]; // se folosesc 2 liste de varfuri
   memcpy(v1, vrf, nv*sizeof(P3D));
// se decupeaza laturile polig. fata de cele 6 planuri in care se afla fețele vol. de decupare
  for(int plan =1 ; plan <= 6; plan+ =2)
  { DecupLaturi(nv, v1, nd, v2, plan);
    DecupLaturi(*nd, v2, &nv, v1, plan+1);
  }
 *nd =nv; *vrfd = new P3D[nv];
  memcpy(*vrfd, v1, nv*sizeof(P3D));
  delete v1; delete v2;
}
```

Algoritmul Weiler – Atherton(1)

Decuparea unui poligon oarecare (poligonul subiect) fata de un poligon oarecare (poligonul de decupare).

Notam cu:

PS- poligonul subiect

PD-poligonul de decupare

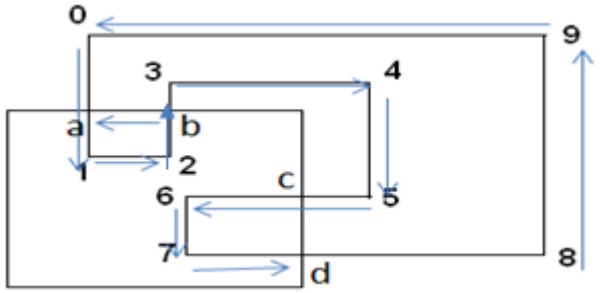
PR- poligonul rezultat – acesta poate fi alcatuit din mai multe cicluri de varfuri (poligoane)

LS – lista de varfuri a PS

LV - lista de varfuri a PR

Vcurent: varful curent din LS (indicele) –

din care începe/se continua parcurgerea conturului PS



- Se parcurg alternativ conturul PS si cel al PD, pastrand sensul de parcurgere (trigonometric).

Cicluri de varfuri rezultate din decupare:

(a, 1, 2, b, a); (c, 6, 7, d, c)

Algoritmul Weiler – Atherton(2)

Se calculeaza punctele de intersectie dintre PS si PD si se clasifica in: **puncte de intrare in PD si puncte de iesire din PD.**

Se initializeaza LV. Se alege un varf al PS exterior PD. Acesta este Vstart (varful din care incepe parcurgerea conturului poligonului PS).

Vcurent ← Vstart; gata = false.

repeta

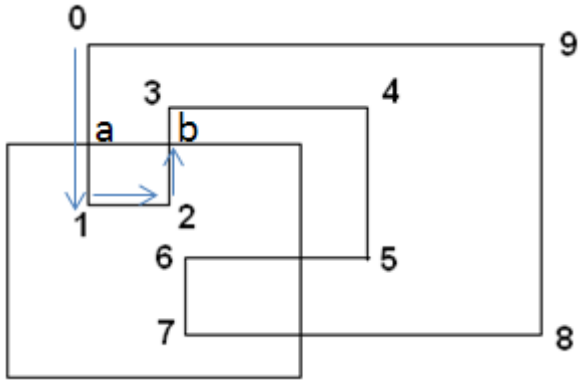
*Se parcurge conturul poligonului PS incepand din Vcurent,
pana cand se ajunge in Vstart
sau intr-un punct Pi de intrare in PD.

daca Vcurent == Vstart

gata = true; break;

altfel

1. Se adauga Pi in LV si se continua parcurgerea pe conturul PS adaugand la LV varfurile PS intalnite pe parcurs, pana intr-un punct ,Pe, de iesire din PD. Se adauga Pe la LV.



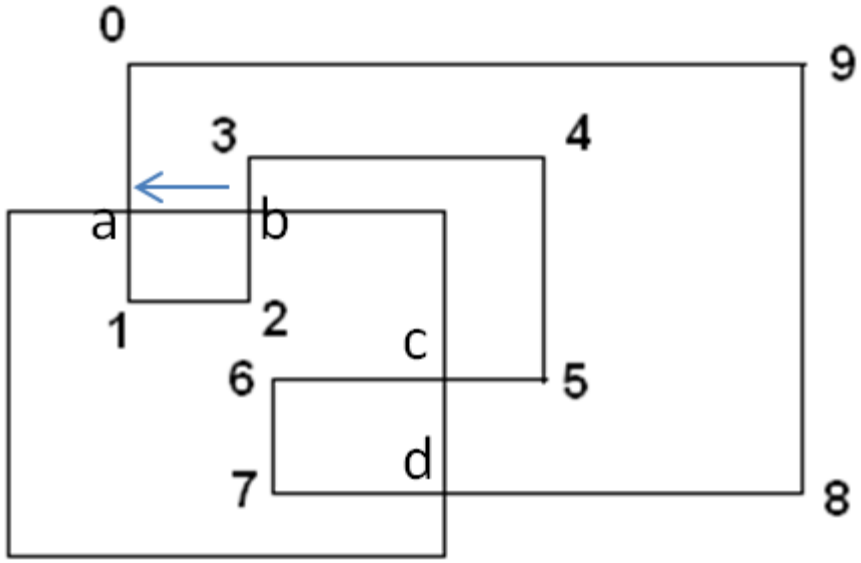
LV: a, 1, 2, b

Algoritmul Weiler – Atherton(3)

2. Vcurent ← vârful exterior al laturii PS care intersecteaza PD in Pe. // vrf. 3

3. Se continua parcurgerea pe conturul PD din Pe pana cand se ajunge intr-un punct, Pi, de intersectie cu PS. Se adauga in LV varfurile PD intalnite pe parcurs si punctul Pi.

4. Daca s-a format un ciclu de varfuri, se memoreaza ciclul de varfuri si se initializeaza LV.
altfel, Vcurent ← Pi (se inlocuieste in LS vrf curent cu Pi).



LV: a, 1, 2, b, a

↓

Ciclul de varfuri c1: a, 1, 2, b, a

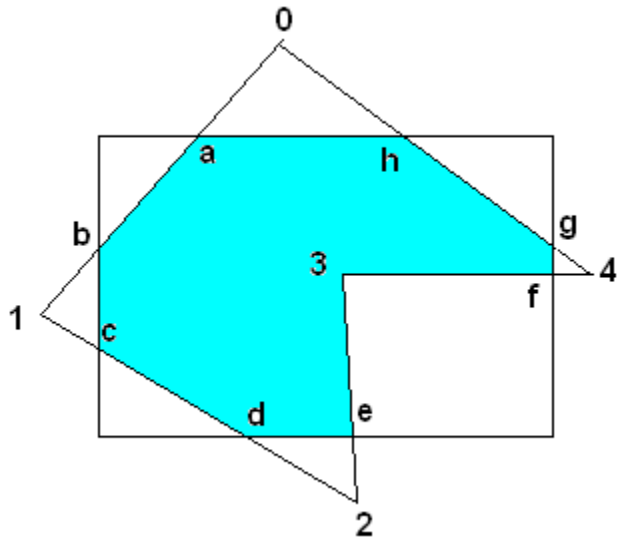
↓

LV: vida

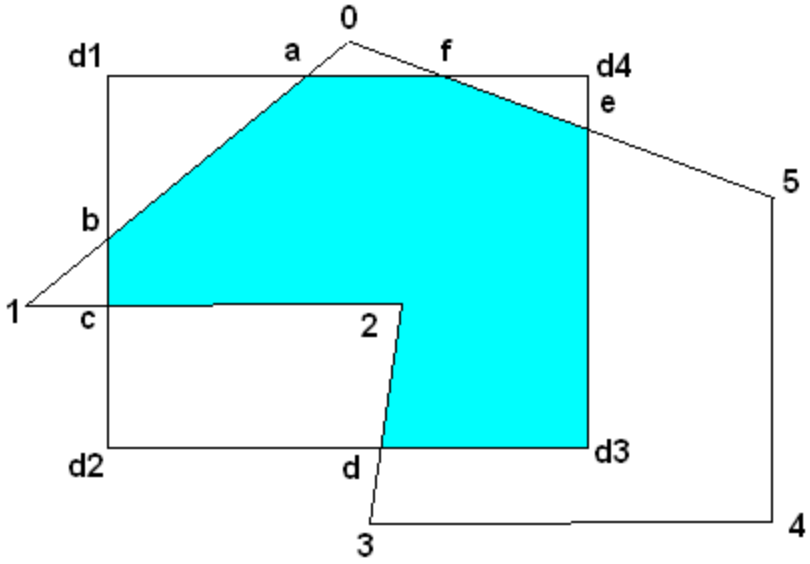
pana(gata)

In iteratiile urmatoare:
-se continua parcurgerea pe PS din varful 3 si se formeaza ciclul de varfuri c2: c, 6, 7, d, c
-se continua parcurgerea pe PS din vârful 8 pana se ajunge in Vstart.

Algoritmul Weiler – Atherton(4)



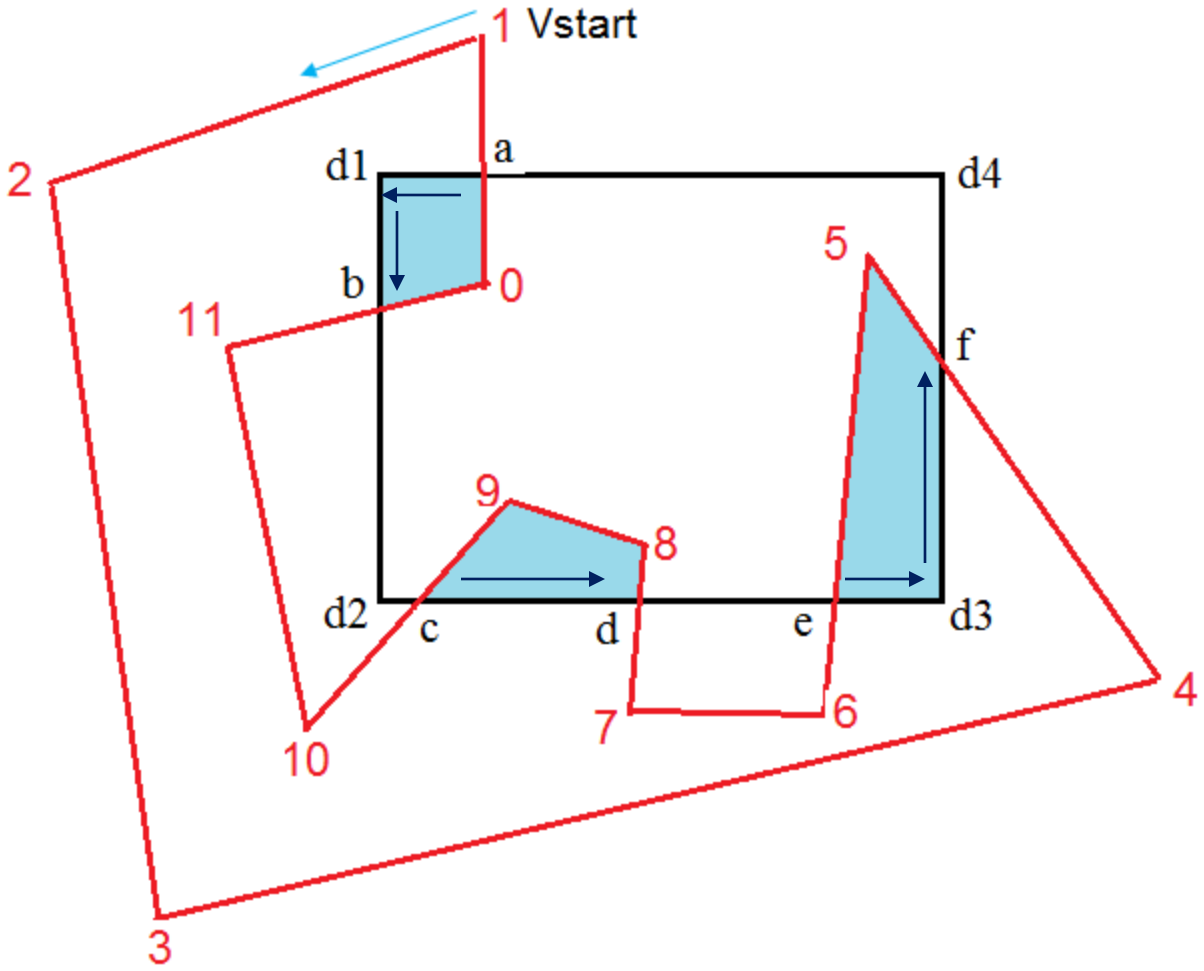
Se porneste din varful 0
LV: a, b (Vcurent ← 1)
a, b, c (Vcurent ← c: se inlocuieste in LS vrf 1 cu c)
a, b, c, d (Vcurent ← 2)
a, b, c, d, e (Vcurent ← e: se inlocuieste in LS vrf 2 cu e)
a, b, c, d, e, 3, f (Vcurent ← 4)
a, b, c, d, e, 3, f, g (Vcurent ← g: se inlocuieste in LS vrf 4 cu g)
a, b, c, d, e, 3, f, g, h (Vcurent ← 0)
a, b, c, d, e, 3, f, g, h, a: s-a format un ciclu de laturi
LV: vida
Se reia din Vcurent: Vcurent == Vstart



Se porneste din varful 0

LV: a, b (Vcurent ← 1)
a, b, c (se inlocuieste in LS vrf. 1 cu c)
a, b, c, 2, d, d3, e (se inlocuieste in LS vrf. 3 cu e)
a, b, c, 2, d, d3, e, f, a

Algoritmul Weiler – Atherton(5)



Vstart ← 1
LV: f,5,e (Vcurent ← 6)
LV: f,5,e,d3, f → s-a format un ciclu de varfuri:
c1: f,5,e,d3, f
- se initializeaza LV
LV: d,8,9,c (Vcurent ← 10)
LV: d,8,9,c,d → s-a format un ciclu de varfuri:
c2: d,8,9,c,d
- se initializeaza LV

LV: b,0,a (Vcurent ←1)
LV: b,0,a,d1,b → s-a format un ciclu de varfuri:
c3: b,0,a,d1,b
- se initializeaza LV
Vcurent == Vstart