Lokakeppnin

- Ljóst er að lokakeppnin verður að vera með breyttu sniði.

Lokakeppnin

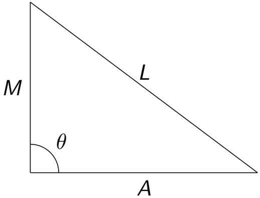
- Ljóst er að lokakeppnin verður að vera með breyttu sniði.
- Hún verður haldin á netinu.

Lokakeppnin

- Ljóst er að lokakeppnin verður að vera með breyttu sniði.
- Hún verður haldin á netinu.
- Við getum því ekki bannað gúggl, en bannað er að dreifa lausnum sínum á meðan á keppninni stendur yfir.

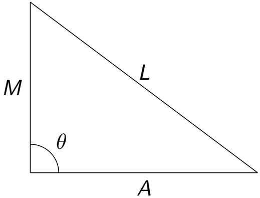
Hornaföll

- Þessi glæra ætti að vera upprifjun fyrir flest ykkar.



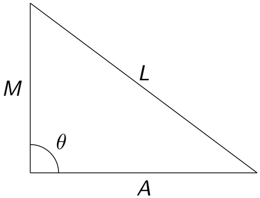
Hornaföll

- Þessi glæra ætti að vera upprifjun fyrir flest ykkar.
- Þríhyrningur er sagður rétthyrndur ef eitt horna hans er 90°.



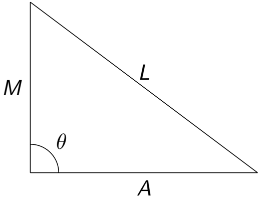
Hornaföll

- Þessi glæra ætti að vera upprifjun fyrir flest ykkar.
- Þríhyrningur er sagður rétthyrndur ef eitt horna hans er 90°.
- Fyrir rétthyrnda þríhyrninga gildir:



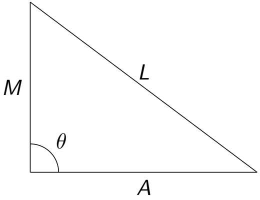
Hornaföll

- Þessi glæra ætti að vera upprifjun fyrir flest ykkar.
- Þríhyrningur er sagður rétthyrndur ef eitt horna hans er 90°.
- Fyrir rétthyrnda þríhyrninga gildir:
  - $\dfrac{A}{L} = \cos\theta$.



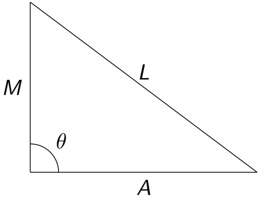
Hornaföll

- Þessi glæra ætti að vera upprifjun fyrir flest ykkar.
- Þríhyrningur er sagður rétthyrndur ef eitt horna hans er 90°.
- Fyrir rétthyrnda þríhyrninga gildir:
  - $\dfrac{A}{L} = \cos\theta$.
  - $\dfrac{M}{L} = \sin\theta$.



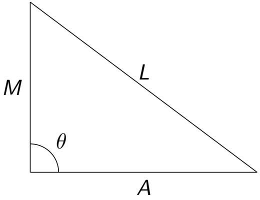
Hornaföll

- Þessi glæra ætti að vera upprifjun fyrir flest ykkar.
- Þríhyrningur er sagður rétthyrndur ef eitt horna hans er 90°.
- Fyrir rétthyrnda þríhyrninga gildir:
  - $\dfrac{A}{L} = \cos\theta$.
  - $\dfrac{M}{L} = \sin\theta$.
  - $\dfrac{M}{A} = \dfrac{M}{L}\dfrac{L}{A} = \dfrac{\sin\theta}{\cos\theta} = \tan\theta$.



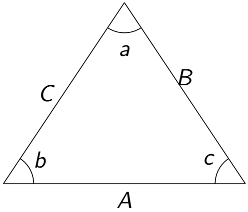
Hornaföll

- Þessi glæra ætti að vera upprifjun fyrir flest ykkar.
- Þríhyrningur er sagður rétthyrndur ef eitt horna hans er 90°.
- Fyrir rétthyrnda þríhyrninga gildir:
  - $\dfrac{A}{L} = \cos\theta$.
  - $\dfrac{M}{L} = \sin\theta$.
  - $\dfrac{M}{A} = \dfrac{M}{L}\dfrac{L}{A} = \dfrac{\sin\theta}{\cos\theta} = \tan\theta$.
- Einnig gildir regla Pýthagorasar, $L^2 = A^2 + M^2$.



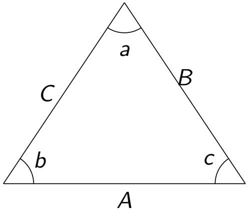
Hornaföll

- Almennar gildir um þríhyrninga:



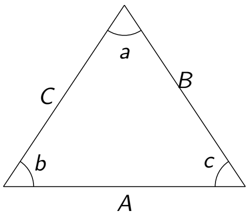
Hornaföll

- Almennar gildir um þríhyrninga:
  - $\dfrac{\sin a}{A} = \dfrac{\sin b}{B} = \dfrac{\sin c}{C}$ (sínus reglan).



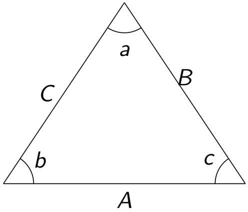
Hornaföll

- Almennar gildir um þríhyrninga:
  - $\dfrac{\sin a}{A} = \dfrac{\sin b}{B} = \dfrac{\sin c}{C}$ (sínus reglan).
  - $A^2 = B^2 + C^2 - 2BC \cos a$ (kósínus reglan)



Hornaföll

- Almennar gildir um þríhyrninga:
  - $\frac{\sin a}{A} = \frac{\sin b}{B} = \frac{\sin c}{C}$ (sínus reglan).
  - $A^2 = B^2 + C^2 - 2BC \cos a$ (kósínus reglan)
- **Æfing:** Sannið reglu Pýthagorasar með kósínus reglunni.

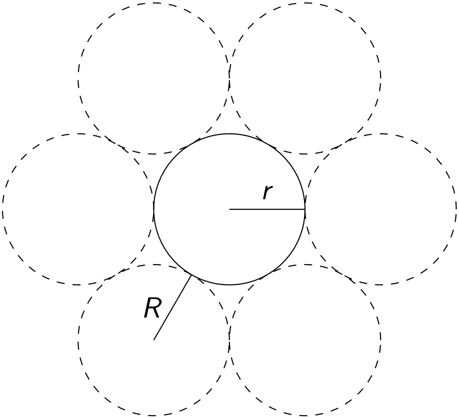


Sýnidæmi: NN and the Optical Illusion - Codeforces

Þér er gefið heiltölu n og rauntölu r. Þú teiknar hring á blað með geilsa r. Þú vilt teikna n jafn stóra hringi í kringum hringinn þinn þannig að þeir skeri hringinn þinn og aðlæga hringi í nákvæmlega einum punkti. Hver þarf geilsi ytri hringjanna að vera.
https://codeforces.com/problemset/problem/1100/C

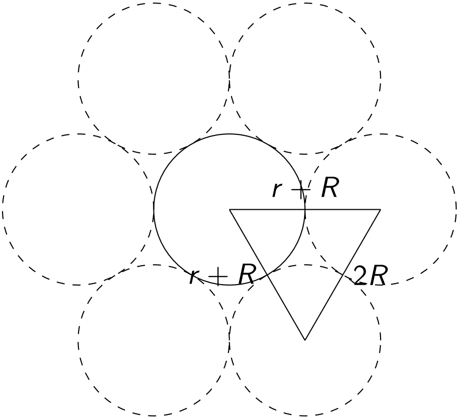
Sýnidæmi: NN and the Optical Illusion - Codeforces

Ef $n = 6$ fæst eftirfarandi mynd, þar sem R er svarið.



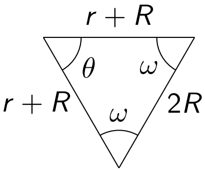
Sýnidæmi: NN and the Optical Illusion - Codeforces

Sjáum að fjarlægðin frá miðjum myndarinnar að miðju ytri hringjanna er $r + R$. Við fáum því eftirfarandi jafnarma þríhyrning.



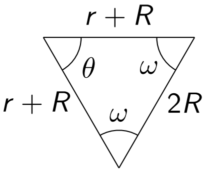
Sýnidæmi: NN and the Optical Illusion - Codeforces

- Hornið θ af síðustu glæru er það eina (ásamt R, að sjálfsögðu) háð n á myndinni, svo almennt þurfum við að finna R út frá eftirfarandi mynd.



Sýnidæmi: NN and the Optical Illusion - Codeforces

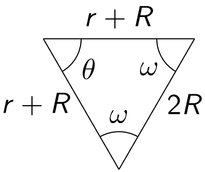
- Hornið θ af síðustu glæru er það eina (ásamt R, að sjálfsögðu) háð n á myndinni, svo almennt þurfum við að finna R út frá eftirfarandi mynd.
- Nú er $\theta = \dfrac{360^\circ}{n}$ og $\omega = \dfrac{180^\circ - \theta}{2}$.



Sýnidæmi: NN and the Optical Illusion - Codeforces

- Sínus reglan gefur okkur svo að

$$\frac{2R}{\sin\theta} = \frac{r+R}{\sin\omega} \Rightarrow 2R\sin\omega - R\sin\theta = r\sin\theta$$
$$\Rightarrow R = \frac{r\sin\theta}{2\sin\omega - \sin\theta}.$$



Tvinntölur

- Skilgreinum mengið $\mathbb{C} := \mathbb{R} \times \mathbb{R}$.

Tvinntölur

- Skilgreinum mengið $\mathbb{C} := \mathbb{R} \times \mathbb{R}$.
- Skilgreinum svo samlagningu á $\mathbb{C}$ þannig að fyrir $(a, b), (c, d) \in \mathbb{C}$ þá

$$(a, b) + (c, d) = (a + c, b + d).$$

Tvinntölur

- Skilgreinum mengið $\mathbb{C} := \mathbb{R} \times \mathbb{R}$.
- Skilgreinum svo samlagningu á $\mathbb{C}$ þannig að fyrir $(a, b), (c, d) \in \mathbb{C}$ þá

$$(a, b) + (c, d) = (a + c, b + d).$$

- Skilgreinum svo margföldun á $\mathbb{C}$ þannig að fyrir $(a, b), (c, d) \in \mathbb{C}$ þá

$$(a, b) \cdot (c, d) = (ac - bd, ad + bc).$$

Tvinntölur

- Skilgreinum mengið $\mathbb{C} := \mathbb{R} \times \mathbb{R}$.
- Skilgreinum svo samlagningu á $\mathbb{C}$ þannig að fyrir $(a, b), (c, d) \in \mathbb{C}$ þá
$$(a, b) + (c, d) = (a + c, b + d).$$
- Skilgreinum svo margföldun á $\mathbb{C}$ þannig að fyrir $(a, b), (c, d) \in \mathbb{C}$ þá
$$(a, b) \cdot (c, d) = (ac - bd, ad + bc).$$
- Við táknum iðulega $(0, 1) \in \mathbb{C}$ með i og $(x, y) \in \mathbb{C}$ með $x + yi$.

Tvinntölur

- Skilgreinum mengið $\mathbb{C} := \mathbb{R} \times \mathbb{R}$.
- Skilgreinum svo samlagningu á $\mathbb{C}$ þannig að fyrir $(a, b), (c, d) \in \mathbb{C}$ þá

$$(a, b) + (c, d) = (a + c, b + d).$$

- Skilgreinum svo margföldun á $\mathbb{C}$ þannig að fyrir $(a, b), (c, d) \in \mathbb{C}$ þá

$$(a, b) \cdot (c, d) = (ac - bd, ad + bc).$$

- Við táknum iðulega $(0, 1) \in \mathbb{C}$ með i og $(x, y) \in \mathbb{C}$ með $x + yi$.
- Stök $\mathbb{C}$ kallast *tvinntölur*.

Tvinntölur

- Ef $z = x + yi \in \mathbb{C}$ þá...

Tvinntölur

- Ef $z = x + yi \in \mathbb{C}$ þá...
  - ...köllum við x *raunhluta* z og y *þverhluta* z.

Tvinntölur

- Ef $z = x + yi \in \mathbb{C}$ þá...
  - ...köllum við x *raunhluta* z og y *þverhluta* z.
  - ...er *lengd* z gefin með $|z| = \sqrt{x^2 + y^2}$.

Tvinntölur

- Ef $z = x + yi \in \mathbb{C}$ þá...
  - ...köllum við x *raunhluta* z og y *þverhluta* z.
  - ...er *lengd* z gefin með $|z| = \sqrt{x^2 + y^2}$.
  - ...köllum við $x - yi$ *samoka* z, táknað $\overline{z}$.

Tvinntölur

- Ef $z = x + yi \in \mathbb{C}$ þá...
  - ...köllum við x *raunhluta* z og y *þverhluta* z.
  - ...er *lengd* z gefin með $|z| = \sqrt{x^2 + y^2}$.
  - ...köllum við $x - yi$ *samoka* z, táknað $\overline{z}$.
  - ...köllum hornið sem (x, y) myndar við jákvæða hluta x-ás í $(0, 0)$ *stefnuhorn* z og táknum það með Arg z.

Kostir tvinntalna í rúmfræði

- Látum nú $x, y \in \mathbb{C}$.

Kostir tvinntalna í rúmfræði

- Látum nú $x, y \in \mathbb{C}$.
- Þá er rúmfræðileg túlkun $x + y$ einfaldlega hliðrun á x um y (eða öfugt).

Kostir tvinntalna í rúmfræði

- Látum nú $x, y \in \mathbb{C}$.
- Þá er rúmfræðileg túlkun $x + y$ einfaldlega hliðrun á x um y (eða öfugt).
- Einnig, ef $|y| = 1$ þá er rúmfræðileg túlkun $x \cdot y$ snúningur á x um $(0, 0)$ um Arg y gráður.

Kostir tvinntalna í rúmfræði

- Látum nú $x, y \in \mathbb{C}$.
- Þá er rúmfræðileg túlkun $x + y$ einfaldlega hliðrun á x um y (eða öfugt).
- Einnig, ef $|y| = 1$ þá er rúmfræðileg túlkun $x \cdot y$ snúningur á x um $(0, 0)$ um Arg y gráður.
- Ef $|x| = r$ og Arg $x = \theta$ þá skrifum við oft $x = re^{i\theta}$.

Kostir tvinntalna í rúmfræði

- Látum nú $x, y \in \mathbb{C}$.
- Þá er rúmfræðileg túlkun $x + y$ einfaldlega hliðrun á x um y (eða öfugt).
- Einnig, ef $|y| = 1$ þá er rúmfræðileg túlkun $x \cdot y$ snúningur á x um $(0, 0)$ um Arg y gráður.
- Ef $|x| = r$ og Arg $x = \theta$ þá skrifum við oft $x = re^{i\theta}$.
- Ef $x = r_1 e^{i\theta_1}$ og $y = r_2 e^{i\theta_2}$ þá er $x \cdot y = r_1 r_2 e^{i(\theta_1 + \theta_2)}$.

Kostir tvinntalna í rúmfræði

- Látum nú $x, y \in \mathbb{C}$.
- Þá er rúmfræðileg túlkun $x + y$ einfaldlega hliðrun á x um y (eða öfugt).
- Einnig, ef $|y| = 1$ þá er rúmfræðileg túlkun $x \cdot y$ snúningur á x um $(0, 0)$ um Arg y gráður.
- Ef $|x| = r$ og Arg $x = \theta$ þá skrifum við oft $x = re^{i\theta}$.
- Ef $x = r_1 e^{i\theta_1}$ og $y = r_2 e^{i\theta_2}$ þá er $x \cdot y = r_1 r_2 e^{i(\theta_1+\theta_2)}$.
- Þetta eru dæmi um hvernig við getum stytt okkur leiðir í rúmfræði með því að nota tvinntölur.

Kostir tvinntalna í rúmfræði

- Látum nú $x, y \in \mathbb{C}$.
- Þá er rúmfræðileg túlkun $x + y$ einfaldlega hliðrun á x um y (eða öfugt).
- Einnig, ef $|y| = 1$ þá er rúmfræðileg túlkun $x \cdot y$ snúningur á x um $(0, 0)$ um Arg y gráður.
- Ef $|x| = r$ og Arg $x = \theta$ þá skrifum við oft $x = re^{i\theta}$.
- Ef $x = r_1 e^{i\theta_1}$ og $y = r_2 e^{i\theta_2}$ þá er $x \cdot y = r_1 r_2 e^{i(\theta_1 + \theta_2)}$.
- Þetta eru dæmi um hvernig við getum stytt okkur leiðir í rúmfræði með því að nota tvinntölur.
- Fleiri (minna augljós) dæmi koma á eftir.

Inngangur

- Við munum bara fjalla um tvívíða rúmfræði í þessum fyrirlestri.

Inngangur

- Við munum bara fjalla um tvívíða rúmfræði í þessum fyrirlestri.
- Þegar leysa þarf þrívíð rúmfræði dæmi er oft góð hugmynd að byrja á að leysa dæmin í tveimur víddum (ef unnt er) og reyna svo að yfirfæra tvívíðu lausnina í þriðju víddina.

Inngangur

- Við munum bara fjalla um tvívíða rúmfræði í þessum fyrirlestri.
- Þegar leysa þarf þrívíð rúmfræði dæmi er oft góð hugmynd að byrja á að leysa dæmin í tveimur víddum (ef unnt er) og reyna svo að yfirfæra tvívíðu lausnina í þriðju víddina.
- Hingað til í námskeiðinu höfum við að mestu fengist við heiltölur og stöku sinnum þurft að vinna með fleytitölur.

Inngangur

- Við munum bara fjalla um tvívíða rúmfræði í þessum fyrirlestri.
- Þegar leysa þarf þrívíð rúmfræði dæmi er oft góð hugmynd að byrja á að leysa dæmin í tveimur víddum (ef unnt er) og reyna svo að yfirfæra tvívíðu lausnina í þriðju víddina.
- Hingað til í námskeiðinu höfum við að mestu fengist við heiltölur og stöku sinnum þurft að vinna með fleytitölur.
- Í rúmfræði er þetta þó öfugt, við vinnum aðallega með fleytitölur og stöku sinnum heiltölur.

Inngangur

- Við munum bara fjalla um tvívíða rúmfræði í þessum fyrirlestri.
- Þegar leysa þarf þrívíð rúmfræði dæmi er oft góð hugmynd að byrja á að leysa dæmin í tveimur víddum (ef unnt er) og reyna svo að yfirfæra tvívíðu lausnina í þriðju víddina.
- Hingað til í námskeiðinu höfum við að mestu fengist við heiltölur og stöku sinnum þurft að vinna með fleytitölur.
- Í rúmfræði er þetta þó öfugt, við vinnum aðallega með fleytitölur og stöku sinnum heiltölur.
- Þegar við notum fleytitölur er mikilvægt að passa að samanburðir er ekki fullkomnir.

Inngangur

- Við munum bara fjalla um tvívíða rúmfræði í þessum fyrirlestri.
- Þegar leysa þarf þrívíð rúmfræði dæmi er oft góð hugmynd að byrja á að leysa dæmin í tveimur víddum (ef unnt er) og reyna svo að yfirfæra tvívíðu lausnina í þriðju víddina.
- Hingað til í námskeiðinu höfum við að mestu fengist við heiltölur og stöku sinnum þurft að vinna með fleytitölur.
- Í rúmfræði er þetta þó öfugt, við vinnum aðallega með fleytitölur og stöku sinnum heiltölur.
- Þegar við notum fleytitölur er mikilvægt að passa að samanburðir er ekki fullkomnir.
- Við látum því duga að tvær tölur sé *nógu* líkar, í vissum skilningi, til að þær séu jafnar.

Fleytitölu samanburðir

```c
// daemi um algengan epsilon-slaka
#define EPS 1e-9

int eq(double a, double b)
{
    return fabs(a - b) < EPS;
}

int neq(double a, double b)
{
    return fabs(a - b) >= EPS;
}
```

Punktar

- Það eru tvær algengar leiðir til að útfæra punkta í plani.

Punktar

- Það eru tvær algengar leiðir til að útfæra punkta í plani.
- Augljósari aðferðin er að skilgreina gagnagrind (`struct`) sem geymir tvær fleytitölur.

Punktar

- Það eru tvær algengar leiðir til að útfæra punkta í plani.
- Augljósari aðferðin er að skilgreina gagnagrind (`struct`) sem geymir tvær fleytitölur.
- Hin aðferðin er að nota innbyggða (í flestum málum) tvinntölu gagnatagið.

Punktar

- Það eru tvær algengar leiðir til að útfæra punkta í plani.
- Augljósari aðferðin er að skilgreina gagnagrind (`struct`) sem geymir tvær fleytitölur.
- Hin aðferðin er að nota innbyggða (í flestum málum) tvinntölu gagnatagið.
- Þó þessi aðgerð gæti verið nokkuð heimulleg þá er hún þægileg og fljótleg í útfærslu.

Notkun á `complex` úr C++ í rúmfræði

- Fallið `real(p)` skilar ofanvarpi p á x-ás.

Notkun á `complex` úr C++ í rúmfræði

- Fallið `real(p)` skilar ofanvarpi p á x-ás.
- Fallið `imag(p)` skilar ofanvarpi p á y-ás.

Notkun á `complex` úr C++ í rúmfræði

- Fallið `real(p)` skilar ofanvarpi p á x-ás.
- Fallið `imag(p)` skilar ofanvarpi p á y-ás.
- Fallið `abs(p)` skilar fjarlægð p frá $(0, 0)$.

Notkun á `complex` úr C++ í rúmfræði

- Fallið `real(p)` skilar ofanvarpi p á x-ás.
- Fallið `imag(p)` skilar ofanvarpi p á y-ás.
- Fallið `abs(p)` skilar fjarlægð p frá $(0, 0)$.
- Fallið `abs(p - q)` skilar fjarlægð milli p og q.

Notkun á `complex` úr C++ í rúmfræði

- Fallið `real(p)` skilar ofanvarpi p á x-ás.
- Fallið `imag(p)` skilar ofanvarpi p á y-ás.
- Fallið `abs(p)` skilar fjarlægð p frá $(0, 0)$.
- Fallið `abs(p - q)` skilar fjarlægð milli p og q.
- Fallið `arg(p)` skilar stefnuhorninu p.

Notkun á `complex` úr C++ í rúmfræði

- Fallið `real(p)` skilar ofanvarpi p á x-ás.
- Fallið `imag(p)` skilar ofanvarpi p á y-ás.
- Fallið `abs(p)` skilar fjarlægð p frá $(0, 0)$.
- Fallið `abs(p - q)` skilar fjarlægð milli p og q.
- Fallið `arg(p)` skilar stefnuhorninu p.
- Fallið `norm(p)` skilar sama og `abs(p)*abs(p)`.

Notkun á `complex` úr C++ í rúmfræði

- Fallið `real(p)` skilar ofanvarpi p á x-ás.
- Fallið `imag(p)` skilar ofanvarpi p á y-ás.
- Fallið `abs(p)` skilar fjarlægð p frá $(0, 0)$.
- Fallið `abs(p - q)` skilar fjarlægð milli p og q.
- Fallið `arg(p)` skilar stefnuhorninu p.
- Fallið `norm(p)` skilar sama og `abs(p)*abs(p)`.
- Fallið `conj(p)` speglar p um x-ás.

Punktar

```
// 1:
typedef struct
{
    double x, y;
} pt;



// 2:
typedef complex<double> pt;
```

Sýnidæmi

Þú byrjar í $(0,0)$ og færð gefnar skipanir. Skipanirnar eru allar einn bókstafur og ein tala. Ef skipunin er `'f' x` gengur þú áfram um x metra, `'b' x` gengur þú aftur á bak um x metra, `'r' x` snýrð þú þér um x radíana til hægri og `'l' x` snýrð þú þér um x radíana til vinstri. Eftir að fylgja öllum þessum skipunum, hversu langt ertu frá $(0,0)$.

Sýnidæmi

- Ef við erum í $p \in \mathbb{C}$ og viljum taka r metra skref í stefnu θ getum við einfaldlega lagt $re^{i\theta}$ við p.

Sýnidæmi

- Ef við erum í $p \in \mathbb{C}$ og viljum taka r metra skref í stefnu θ getum við einfaldlega lagt $re^{i\theta}$ við p.
- Hvert við snúum í upphafi skiptir ekki mál því það hefur ekki áhrif á fjarlægðinni til $(0, 0)$.

Sýnidæmi

```
int main()
{
    double x, r = 0.0;
    pt p(0.0, 0.0);
    int n = get_int();
    while (n-- != 0)
    {
        char c = getchar();
        x = get_int();
        if (c == 'f')      p += x*exp(pt(0.0, r));
        else if (c == 'b') p -= x*exp(pt(0.0, r));
        else if (c == 'l') r += x;
        else if (c == 'r') r -= x;
    }
    printf("%.8f\n", abs(p));
    return 0;
}
```

Línur

- Samkvæmt fyrstu frumsendu rúmfræðinnar skilgreina tveir ólíkir punktar nákvæmlega eina línu.

Línur

- Samkvæmt fyrstu frumsendu rúmfræðinnar skilgreina tveir ólíkir punktar nákvæmlega eina línu.
- Svo við getum einfaldlega sagt að lína sé tveir ólíkir punktar.

Línur

- Samkvæmt fyrstu frumsendu rúmfræðinnar skilgreina tveir ólíkir punktar nákvæmlega eina línu.
- Svo við getum einfaldlega sagt að lína sé tveir ólíkir punktar.
- Helsti ókostur þessa aðferðar er að sama línan getur verið skilgreint með mismunandi pörum af punktum.

Línur

- Samkvæmt fyrstu frumsendu rúmfræðinnar skilgreina tveir ólíkir punktar nákvæmlega eina línu.
- Svo við getum einfaldlega sagt að lína sé tveir ólíkir punktar.
- Helsti ókostur þessa aðferðar er að sama línan getur verið skilgreint með mismunandi pörum af punktum.
- Stundum hentar betur að skilgreina línu með skurðpunkt við y-ás og hallatölu.

Línur

- Samkvæmt fyrstu frumsendu rúmfræðinnar skilgreina tveir ólíkir punktar nákvæmlega eina línu.
- Svo við getum einfaldlega sagt að lína sé tveir ólíkir punktar.
- Helsti ókostur þessa aðferðar er að sama línan getur verið skilgreint með mismunandi pörum af punktum.
- Stundum hentar betur að skilgreina línu með skurðpunkt við y-ás og hallatölu.
- Þá er einfaldara að bera saman línur en það þarf að höndla sérstaklega línur samsíða y-ás.

Skurðpunktur lína

- Ef tvær línur eru ekki samsíða þá skerast þær í nákvæmlega einum punkti.

Skurðpunktur lína

- Ef tvær línur eru ekki samsíða þá skerast þær í nákvæmlega einum punkti.
- Við þurfum oft að finna þennan punkt.

Skurðpunktur lína

- Ef tvær línur eru ekki samsíða þá skerast þær í nákvæmlega einum punkti.
- Við þurfum oft að finna þennan punkt.
- Gefum okkur tvær línur $\{(x, y) : ax + by = c\}$ og $\{(x, y) : dx + ey = f\}$.

Skurðpunktur lína

- Ef tvær línur eru ekki samsíða þá skerast þær í nákvæmlega einum punkti.
- Við þurfum oft að finna þennan punkt.
- Gefum okkur tvær línur $\{(x, y) : ax + by = c\}$ og $\{(x, y) : dx + ey = f\}$.
- Gerum ráð fyrir að línurnar séu ekki samsíða.

Skurðpunktur lína

- Ef tvær línur eru ekki samsíða þá skerast þær í nákvæmlega einum punkti.
- Við þurfum oft að finna þennan punkt.
- Gefum okkur tvær línur $\{(x, y) : ax + by = c\}$ og $\{(x, y) : dx + ey = f\}$.
- Gerum ráð fyrir að línurnar séu ekki samsíða.
- Skurðpunkturinn fæst þá greinilega með því að leysa jöfnuhneppið

$$\left(\begin{array}{cc|c} a & b & c \\ d & e & f \end{array}\right)$$

Skurðpunktur lína

$$\begin{pmatrix} a & b \\ d & e \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} c \\ f \end{pmatrix}$$

Skurðpunktur lína

$$\begin{pmatrix} a & b \\ d & e \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} c \\ f \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a & b \\ d & e \end{pmatrix}^{-1} \begin{pmatrix} c \\ f \end{pmatrix}$$

Skurðpunktur lína

$$\begin{pmatrix} a & b \\ d & e \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} c \\ f \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a & b \\ d & e \end{pmatrix}^{-1} \begin{pmatrix} c \\ f \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{ae - bd} \begin{pmatrix} e & -b \\ -d & a \end{pmatrix} \begin{pmatrix} c \\ f \end{pmatrix}$$

Skurðpunktur lína

$$\begin{pmatrix} a & b \\ d & e \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} c \\ f \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a & b \\ d & e \end{pmatrix}^{-1} \begin{pmatrix} c \\ f \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{ae - bd} \begin{pmatrix} e & -b \\ -d & a \end{pmatrix} \begin{pmatrix} c \\ f \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{ae - bd} \begin{pmatrix} ce - bf \\ af - cd \end{pmatrix}$$

Skurðpunktur lína

$$\begin{pmatrix} a & b \\ d & e \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} c \\ f \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a & b \\ d & e \end{pmatrix}^{-1} \begin{pmatrix} c \\ f \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{ae - bd} \begin{pmatrix} e & -b \\ -d & a \end{pmatrix} \begin{pmatrix} c \\ f \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{ae - bd} \begin{pmatrix} ce - bf \\ af - cd \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \frac{ce-bf}{ae-bd} \\ \frac{af-cd}{ae-bd} \end{pmatrix}$$

Línustrik

- Af augljósum ástæðum sést að best er að skilgreina línustrik sem par punkta, nánar tiltekið endapunkta línustriksins.

Línustrik

- Af augljósum ástæðum sést að best er að skilgreina línustrik sem par punkta, nánar tiltekið endapunkta línustriksins.
- Markmið okkar í þessum hluta af fyrirlestrinu verður að útfæra fall sem finnur fjarlægð milli línustrika.

Línustrik

- Af augljósum ástæðum sést að best er að skilgreina línustrik sem par punkta, nánar tiltekið endapunkta línustriksins.
- Markmið okkar í þessum hluta af fyrirlestrinu verður að útfæra fall sem finnur fjarlægð milli línustrika.
- Við munum byrja á að útfæra góð hjálparföll sem nýtast meðal annars í að finna þessa fjarlægð, en eru einnig hentug í öðrum dæmum.

Línustrik

- Við munum útfæra:

Línustrik

- Við munum útfæra:
  - Fall sem skoðar hvort tvö bil skerist (`bxb(...)`).

Línustrik

- Við munum útfæra:
  - Fall sem skoðar hvort tvö bil skerist (`bxb(...)`).
  - Fall sem skoðar hvort línustrik skerist (`lxl(...)`).

Línustrik

- Við munum útfæra:
  - Fall sem skoðar hvort tvö bil skerist (`bxb(...)`).
  - Fall sem skoðar hvort línustrik skerist (`lxl(...)`).
  - Fall sem finnur stystu fjarlægð punkts og línustriks (`p2l(...)`).

Línustrik

- Við munum útfæra:
  - Fall sem skoðar hvort tvö bil skerist (`bxb(...)`).
  - Fall sem skoðar hvort línustrik skerist (`lxl(...)`).
  - Fall sem finnur stystu fjarlægð punkts og línustriks (`p2l(...)`).
  - Fall sem finnur stystu fjarlægð tveggja línustrika (`l2l(...)`).

Skurður bila

- Þegar við viljum skoða skurð tveggja bila nægir okkur að skoða hvort annar endapunktur bils er í hinu bilinu.

Skurður bila

- Þegar við viljum skoða skurð tveggja bila nægir okkur að skoða hvort annar endapunktur bils er í hinu bilinu.

```
int bxb(double a, double b, double c, double d)
{ // Skerast [a, b] og [c, d]?
    if (a > b) return bxb(b, a, c, d);
    if (c > d) return bxb(a, b, d, c);
    return fmax(a, c) <= fmin(b, d);
}
```

Skurður tveggja línustrika

- Þessi glæra gæti verið nokkuð dularfull en hún mun verða skýrari seinna í þessum fyrirlestri.

Skurður tveggja línustrika

- Þessi glæra gæti verið nokkuð dularfull en hún mun verða skýrari seinna í þessum fyrirlestri.
- Við munum skoða áttunina á þríhyrningunum sem við getum myndað með endapunktum línustrikanna.

Skurður tveggja línustrika

- Þessi glæra gæti verið nokkuð dularfull en hún mun verða skýrari seinna í þessum fyrirlestri.
- Við munum skoða áttunina á þríhyrningunum sem við getum myndað með endapunktum línustrikanna.
- Látum a, b, c, d vera endapunkta línustrikanna.

Skurður tveggja línustrika

- Þessi glæra gæti verið nokkuð dularfull en hún mun verða skýrari seinna í þessum fyrirlestri.
- Við munum skoða áttunina á þríhyrningunum sem við getum myndað með endapunktum línustrikanna.
- Látum a, b, c, d vera endapunkta línustrikanna.
- Ef þríhyningurinn stikaður með $\langle a, b, c, a \rangle$ hefur öfuga áttun miðað við $\langle a, b, d, a \rangle$ þá liggja punktarnir c og d sitthvoru megin við línustrikið $\langle a, b \rangle$.

Skurður tveggja línustrika

- Þessi glæra gæti verið nokkuð dularfull en hún mun verða skýrari seinna í þessum fyrirlestri.
- Við munum skoða áttunina á þríhyrningunum sem við getum myndað með endapunktum línustrikanna.
- Látum a, b, c, d vera endapunkta línustrikanna.
- Ef þríhyningurinn stikaður með $\langle a, b, c, a \rangle$ hefur öfuga áttun miðað við $\langle a, b, d, a \rangle$ þá liggja punktarnir c og d sitthvoru megin við línustrikið $\langle a, b \rangle$.
- Við þurfum líka að ganga úr skugga um ferhyrningarnir sem endapunktarnir mynda skerast.

Skurður tveggja línustrika

- Það eru leiðinleg sértilfelli þegar þrír af endapunktunum liggja á sömu línunni.

Skurður tveggja línustrika

- Það eru leiðinleg sértilfelli þegar þrír af endapunktunum liggja á sömu línunni.
- Ég mun eftirláta ykkur að laga þessi sértilfelli, því þar sem við höfum aðallega áhuga á að finna fjarlægð línubila nægir að segja að línustrikin skerist ekki í þessu sértilfelli.

Skurður tveggja línustrika

- Það eru leiðinleg sértilfelli þegar þrír af endapunktunum liggja á sömu línunni.
- Ég mun eftirláta ykkur að laga þessi sértilfelli, því þar sem við höfum aðallega áhuga á að finna fjarlægð línubila nægir að segja að línustrikin skerist ekki í þessu sértilfelli.
- Þetta mun skýrast betur á eftir.

Skurður tveggja línustrika

```
int ccw(pt a, pt b, pt c)
{
    double f = cimag((c - a)/(b - a));
    if (fabs(f) < EPS) return 0;
    if (f < EPS) return -1;
    return 1;
}

int lxl(pt a, pt b, pt c, pt d)
{ // Skerast <a, b> og <c, d>?
    int a1 = ccw(a, b, c), a2 = ccw(a, b, d),
        a3 = ccw(c, d, a), a4 = ccw(c, d, b);
    if (a1*a2*a3*a4 == 0) return 0;
    if (a1*a2 != -1 || a3*a4 != -1) return 0;
    return bxb(creal(a), creal(b), creal(c), creal(d))
        && bxb(cimag(a), cimag(b), cimag(c), cimag(d));
}
```

Fjarlægð punkts og línustriks

- Hér hefst fjörið.

Fjarlægð punkts og línustriks

- Hér hefst fjörið.
- Látum línustrikið vera $\langle (x_0, y_0), (x_1, y_1) \rangle$ og punktinn (x, y).

Fjarlægð punkts og línustriks

- Hér hefst fjörið.
- Látum línustrikið vera $\langle (x_0, y_0), (x_1, y_1) \rangle$ og punktinn (x, y).
- Við getum þá stikað línustrikið með $l(t) = (x_0 + t \cdot (x_1 - x_0), y_0 + t \cdot (y_1 - y_0)), t \in [0, 1]$.

Fjarlægð punkts og línustriks

- Hér hefst fjörið.
- Látum línustrikið vera $\langle (x_0, y_0), (x_1, y_1) \rangle$ og punktinn (x, y).
- Við getum þá stikað línustrikið með $l(t) = (x_0 + t \cdot (x_1 - x_0), y_0 + t \cdot (y_1 - y_0)), t \in [0, 1]$.
- Látum d tákna Evklíðsku firðina. Við munum nú lágmarka d^2 til að auðvelda deildunina.

Fjarlægð punkts og línustriks

- Látum $f(t) = d(l(t), (x, y))^2$.

Fjarlægð punkts og línustriks

- Látum $f(t) = d(l(t), (x, y))^2$.
- Við fáum enn fremur að

$$
\begin{aligned}
f(t) =& (x_0 + t \cdot (x_1 - x_0) - x)^2 + (y_0 + t \cdot (y_1 - y_0) - y)^2 \\
=& x_0^2 + t^2(x_1 - x_0)^2 + x^2 + 2x_0 t(x_1 - x_0) - 2xx_0 \\
& -2tx(x_1 - x_0) + y_0^2 + t^2(y_1 - y_0)^2 + y^2 + 2y_0 t(y_1 - y_0) \\
& -2yy_0 - 2ty(y_1 - y_0).
\end{aligned}
$$

Fjarlægð punkts og línustriks

- Látum $f(t) = d(l(t),(x,y))^2$.
- Við fáum enn fremur að

$$\begin{aligned}
f(t) =& (x_0 + t\cdot(x_1 - x_0) - x)^2 + (y_0 + t\cdot(y_1 - y_0) - y)^2\\
=& x_0^2 + t^2(x_1 - x_0)^2 + x^2 + 2x_0 t(x_1 - x_0) - 2xx_0\\
& -2tx(x_1 - x_0) + y_0^2 + t^2(y_1 - y_0)^2 + y^2 + 2y_0 t(y_1 - y_0)\\
& -2yy_0 - 2ty(y_1 - y_0).
\end{aligned}$$

- Deildum nú f og fáum

$$\begin{aligned}
f'(t) =& 2t(x_1 - x_0)^2 + 2x_0(x_1 - x_0) - 2x(x_1 - x_0)\\
& +2t(y_1 - y_0)^2 + 2y_0(y_1 - y_0) - 2y(y_1 - y_0).
\end{aligned}$$

Fjarlægð punkts og línustriks

- Metum nú í 0 óg fáum

$$
\begin{aligned}
& f'(t_0) = 0 \\
\Rightarrow & t_0(x_1 - x_0)^2 + x_0(x_1 - x_0) - x(x_1 - x_0) \\
& + t_0(y_1 - y_0)^2 + y_0(y_1 - y_0) - y(y_1 - y_0) = 0 \\
\Rightarrow & t_0 = \frac{x(x_1 - x_0) + y(y_1 - y_0) - x_0(x_1 - x_0) - y_0(y_1 - y_0)}{(y_1 - y_0)^2 + (x_1 - x_0)^2} \\
\Rightarrow & t_0 = \frac{(x - x_0)(x_1 - x_0) + (y - y_0)(y_1 - y_0)}{(y_1 - y_0)^2 + (x_1 - x_0)^2}.
\end{aligned}
$$

Fjarlægð punkts og línustriks

- Við erum nú búin að finna þann punkt á línunni gegnum (x_0, y_0) og (x_1, y_1) sem er næstur (x, y).

Fjarlægð punkts og línustriks

- Við erum nú búin að finna þann punkt á línunni gegnum (x_0, y_0) og (x_1, y_1) sem er næstur (x, y).
- Þar sem línustrikið er stikað af t þegar $t \in [0, 1]$ þá er þessi punktur á línustrikinu ef $t_0 \in [0, 1]$.

Fjarlægð punkts og línustriks

- Við erum nú búin að finna þann punkt á línunni gegnum (x_0, y_0) og (x_1, y_1) sem er næstur (x, y).
- Þar sem línustrikið er stikað af t þegar $t \in [0, 1]$ þá er þessi punktur á línustrikinu ef $t_0 \in [0, 1]$.
- Ef svo er ekki nægir okkur að skoða fjarlægð (x, y) til endapunktana (x_0, y_0) og (x_1, y_1).

Fjarlægð punkts og línustriks

- Við erum nú búin að finna þann punkt á línunni gegnum (x_0, y_0) og (x_1, y_1) sem er næstur (x, y).
- Þar sem línustrikið er stikað af t þegar $t \in [0, 1]$ þá er þessi punktur á línustrikinu ef $t_0 \in [0, 1]$.
- Ef svo er ekki nægir okkur að skoða fjarlægð (x, y) til endapunktana (x_0, y_0) og (x_1, y_1).

```
double p2l(pt p, pt l1, pt l2)
{ // Finnur fjarlægð frá punkti í línustrik.
    p = (p - l1)*cexp(-I*carg(l2 - l1));
    l2 -= l1;
    if (-EPS < creal(p) && creal(p) < cabs(l2) + EPS)
        return fabs(cimag(p));
    return fmin(cabs(p), cabs(p - cabs(l2)));
}
```

Fjarlægð milli tveggja línustrika

- Gefum okkur línustrikin $\langle a, b\rangle$ og $\langle c, d\rangle$.

Fjarlægð milli tveggja línustrika

- Gefum okkur línustrikin $\langle a, b \rangle$ og $\langle c, d \rangle$.
- Ef þau skerast þá er fjarlægðin á milli þeirra 0.

Fjarlægð milli tveggja línustrika

- Gefum okkur línustrikin $\langle a, b\rangle$ og $\langle c, d\rangle$.
- Ef þau skerast þá er fjarlægðin á milli þeirra 0.
- Gerum því ráð fyrir að þau skerist ekki.

Fjarlægð milli tveggja línustrika

- Gefum okkur línustrikin $\langle a, b\rangle$ og $\langle c, d\rangle$.
- Ef þau skerast þá er fjarlægðin á milli þeirra 0.
- Gerum því ráð fyrir að þau skerist ekki.
- Þá er bersýnilega fjarlægð línustrikana minnst á milli a og $\langle c, d\rangle$, b og $\langle c, d\rangle$, c og $\langle a, b\rangle$ eða d og $\langle a, b\rangle$.

Fjarlægð milli tveggja línustrika

- Ég hef ekki minnst á það hingað til, en við höfum gert ráð fyrir að endapunktar línustrikana séu mismunandi.

Fjarlægð milli tveggja línustrika

- Ég hef ekki minnst á það hingað til, en við höfum gert ráð fyrir að endapunktar línustrikana séu mismunandi.
- Ef $a = b$ og $c = d$ þá er fjarlægð línustrikana fjarlægðin milli a og c.

Fjarlægð milli tveggja línustrika

- Ég hef ekki minnst á það hingað til, en við höfum gert ráð fyrir að endapunktar línustrikana séu mismunandi.
- Ef $a = b$ og $c = d$ þá er fjarlægð línustrikana fjarlægðin milli a og c.
- Ef $a = b$ og $c \neq d$ þá er fjarlægð línustrikana fjarlægðin milli a og $\langle c, d \rangle$.

Fjarlægð milli tveggja línustrika

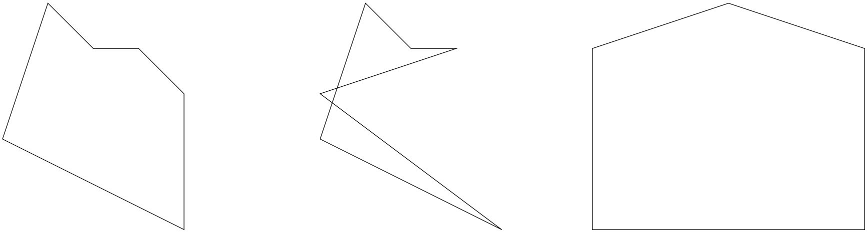
- Ég hef ekki minnst á það hingað til, en við höfum gert ráð fyrir að endapunktar línustrikana séu mismunandi.
- Ef $a = b$ og $c = d$ þá er fjarlægð línustrikana fjarlægðin milli a og c.
- Ef $a = b$ og $c \neq d$ þá er fjarlægð línustrikana fjarlægðin milli a og $\langle c, d \rangle$.
- Ef $a \neq b$ og $c = d$ þá er fjarlægð línustrikana fjarlægðin milli c og $\langle a, b \rangle$.

Fjarlægð milli tveggja línustrika

```
double l2l(pt a1, pt a2, pt b1, pt b2)
{ // Finnur fjarlægð frá línustriki í línustrik.
    if (cabs(a1 - a2) < EPS && cabs(b1 - b2) < EPS)
        return cabs(a1 - b1);
    if (cabs(a1 - a2) < EPS) return p2l(a1, b1, b2);
    if (cabs(b1 - b2) < EPS) return p2l(b1, a1, a2);
    if (lxl(a1, a2, b1, b2)) return 0.0;
    return fmin(fmin(p2l(a1, b1, b2), p2l(a2, b1, b2)),
                fmin(p2l(b1, a1, a2), p2l(b2, a1, a2)));
}
```

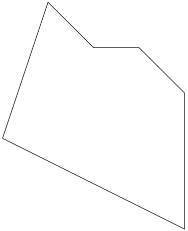
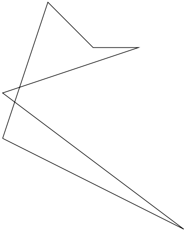
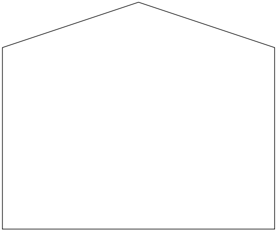
Marghyrningar

- *Marghyrningur* er samfelldur, lokaður ferill í plani sem samanstendur af beinum línustrikum.



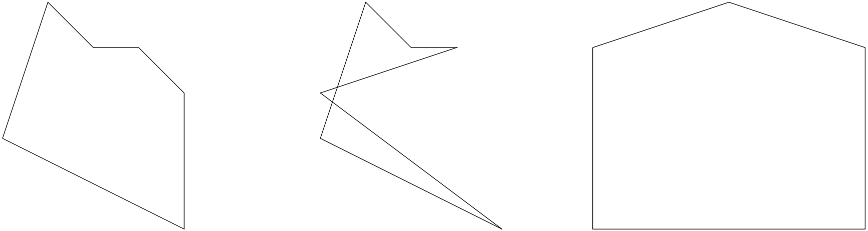
Marghyrningar

- *Marghyrningur* er samfelldur, lokaður ferill í plani sem samanstendur af beinum línustrikum.
- Ef ferillinn er einfaldur þá kallast marghyrningurinn *einfaldur*.

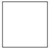
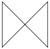




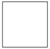
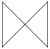
Marghyrningar

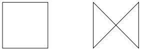
- *Marghyrningur* er samfelldur, lokaður ferill í plani sem samanstendur af beinum línustrikum.
- Ef ferillinn er einfaldur þá kallast marghyrningurinn *einfaldur*.
- Marghyrningur eru sagður vera *kúptur* ef sérhver beina lína, sem er ekki samsíða hlið marghyrningsins, dregin gegnum marghyrninginn sker mest tvo punkta á marghyrningnum.



- Þegar við viljum tákna marghyrning í tölvu notum við einfaldlega hornpunkta hans.

- Þegar við viljum tákna marghyrning í tölvu notum við einfaldlega hornpunkta hans.
- Röð punktanna skiptir máli.  

- Þegar við viljum tákna marghyrning í tölvu notum við einfaldlega hornpunkta hans.
- Röð punktanna skiptir máli.  
- Til þæginda geymum við oft einn punkt tvisvar, nánar tiltekið er fremsti og aftasti punkturinn eins.

- Þegar við viljum tákna marghyrning í tölvu notum við einfaldlega hornpunkta hans.
- Röð punktanna skiptir máli. 
- Til þæginda geymum við oft einn punkt tvisvar, nánar tiltekið er fremsti og aftasti punkturinn eins.
- Þetta er því við höfum oft meira áhuga á línustrikunum milli hornpuntkanna heldur en hornpunktunum sjálfum.

- Þar sem marghyrningar er mjög vinsælir í keppnum munum við fara í nokkur atriði sem er gott að kunna.

Ummál marghyrnings

- Ummála marghyrnings er einfalt að reikna í línulegum tíma.

Ummál marghyrnings

- Ummála marghyrnings er einfalt að reikna í línulegum tíma.
- Maður leggur einfaldlega saman allar hliðarlengdirnar.

Ummál marghyrnings

- Ummála marghyrnings er einfalt að reikna í línulegum tíma.
- Maður leggur einfaldlega saman allar hliðarlengdirnar.

```
double ummal(polygon p)
{ // p[0] == p[n - 1]
    int i;
    double r = 0.0;
    for (i = 0; i < p.size() - 1; i++)
        r = r + abs(p[i] - p[i + 1]);
    return r;
}
```

Flatarmál marghyrnings

- Ummál marghyrninga er þó ekki jafnt algengt í keppnum og flatarmál marghyrninga.

Flatarmál marghyrnings

- Ummál marghyrninga er þó ekki jafnt algengt í keppnum og flatarmál marghyrninga.
- Það er einnig auðvelt að reikna flatarmálið í línulegum tíma, þó það sé ekki endilega augljóst að þetta skili flatarmálinu.

Flatarmál marghyrnings

- Ummál marghyrninga er þó ekki jafnt algengt í keppnum og flatarmál marghyrninga.
- Það er einnig auðvelt að reikna flatarmálið í línulegum tíma, þó það sé ekki endilega augljóst að þetta skili flatarmálinu.
- Fyrir áhugasama er hægt að nota setningu Green til að leiða út eftirfarandi forritsbút.

Flatarmál marghyrnings

```
double flatarmal(polygon &p)
{
    int i;
    double r = 0.0;
    for (i = 0; i < p.size() - 1; i++)
        r = r + real(p[i])*imag(p[i + 1]) - real(p[i + 1])*imag(p[i]);
    return fabs(0.5*r);
}
```

Flatarmál marghyrnings

- Við þurfum að skila tölugildinu því við getum fengið neikvætt flatarmál.

Flatarmál marghyrnings

- Við þurfum að skila tölugildinu því við getum fengið neikvætt flatarmál.
- Neikvætt flatarmál hljómar kannski furðulega en það fellur eðlilega úr sönnun summunar ef notast er við setningu Green.

Flatarmál marghyrnings

- Við þurfum að skila tölugildinu því við getum fengið neikvætt flatarmál.
- Neikvætt flatarmál hljómar kannski furðulega en það fellur eðlilega úr sönnun summunar ef notast er við setningu Green.
- Til að nota hana þarf að reikna ferilheildi og útkoman úr ferilheildum skiptir um formerki þegar breytt er um átt stikunar ferilsins.

Flatarmál marghyrnings

- Við þurfum að skila tölugildinu því við getum fengið neikvætt flatarmál.
- Neikvætt flatarmál hljómar kannski furðulega en það fellur eðlilega úr sönnun summunar ef notast er við setningu Green.
- Til að nota hana þarf að reikna ferilheildi og útkoman úr ferilheildum skiptir um formerki þegar breytt er um átt stikunar ferilsins.
- Þetta þýðir að formerki `r` eftir forlykkjuna er jákvætt ef punktar `p` eru gefnir rangsælis og neikvætt ef þeir eru gefnir réttsælis.

Punktur í marghyrning

- Að ákvarða hvort punktur sé inni í marghyrning (e. the point in polygon problem) er algent undirvandamál í rúmfræði dæmum.

Punktur í marghyrning

- Að ákvarða hvort punktur sé inni í marghyrning (e. the point in polygon problem) er algent undirvandamál í rúmfræði dæmum.
- Aðallega er gengist við tvær aðferðir til að leysa slík dæmi, sú fyrri er að nota geislarakningu (e. raytracing) og hin er að reikna summu aðliggjandi horna marghyrningsins miðað við punktinn.

Geislarakning

- Við getum dregið geisla frá punktinum sem við viljum skoða í einhverja átt og talið hversu oft við skerum jaðar marghyrningsins.

Geislarakning

- Við getum dregið geisla frá punktinum sem við viljum skoða í einhverja átt og talið hversu oft við skerum jaðar marghyrningsins.
- Ef við erum fyrir utan marghyrninginn og skerum jaðarinn erum við inni í honum, en ef við erum fyrir innan og skerum jaðarinn erum við fyrir utan (þetta er í raun skilgreining á því hvenær geislinn *sker* marghyrninginn).

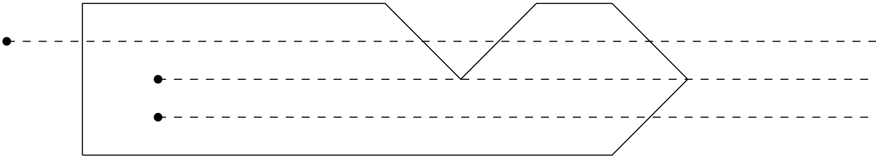
Geislarakning

- Við getum dregið geisla frá punktinum sem við viljum skoða í einhverja átt og talið hversu oft við skerum jaðar marghyrningsins.
- Ef við erum fyrir utan marghyrninginn og skerum jaðarinn erum við inni í honum, en ef við erum fyrir innan og skerum jaðarinn erum við fyrir utan (þetta er í raun skilgreining á því hvenær geislinn *sker* marghyrninginn).
- Svo ef við skerum jaðarinn slétt tölu sinnum er punkturinn fyrir innan, og annars fyrir utan (Setning Jordan).

Geislarakning

- Við getum dregið geisla frá punktinum sem við viljum skoða í einhverja átt og talið hversu oft við skerum jaðar marghyrningsins.
- Ef við erum fyrir utan marghyrninginn og skerum jaðarinn erum við inni í honum, en ef við erum fyrir innan og skerum jaðarinn erum við fyrir utan (þetta er í raun skilgreining á því hvenær geislinn *sker* marghyrninginn).
- Svo ef við skerum jaðarinn slétt tölu sinnum er punkturinn fyrir innan, og annars fyrir utan (Setning Jordan).
- Við getum látið geislann vera línustrik, nógu langt til að vera út fyrir marghyrninginn, og notað síðan `lxl` til að ákvarða í línulegum tíma hversu oft geislinn sker marghyrninginn.

Geislarakning



Geislarakning

- Þessi aðferð er með nokkur sértilfelli sem gerir hana óþægilega í útfærslu.

Geislarakning

- Þessi aðferð er með nokkur sértilfelli sem gerir hana óþægilega í útfærslu.
- Öll sértilfellin eiga það sameiginlegt að vera þegar geislinn sker endapunkta línustrika marghyrningsins.

Geislarakning

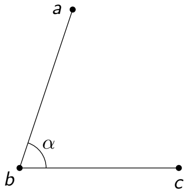
- Þessi aðferð er með nokkur sértilfelli sem gerir hana óþægilega í útfærslu.
- Öll sértilfellin eiga það sameiginlegt að vera þegar geislinn sker endapunkta línustrika marghyrningsins.
- Ef marghyrningurinn er kúptur er nokkuð auðvelt að eiga við þessi sértilfelli, en það gildir ekki í flestum dæmum.

Geislarakning

- Þessi aðferð er með nokkur sértilfelli sem gerir hana óþægilega í útfærslu.
- Öll sértilfellin eiga það sameiginlegt að vera þegar geislinn sker endapunkta línustrika marghyrningsins.
- Ef marghyrningurinn er kúptur er nokkuð auðvelt að eiga við þessi sértilfelli, en það gildir ekki í flestum dæmum.
- Þessi aðferð verður því ekki úrfærð hér.

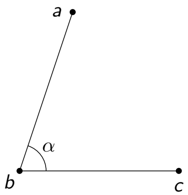
Afstæð hornasumma

- Látum p_i, $i < n$ tákna hornpunkta marghyrnings, q einhvern punkt, $\alpha(a, b, c)$ vera hornið milli a, b og c og $\beta(a, b, c)$ vera 1 ef brotna línustrikið $\langle a, b, c \rangle$ „beygir" til vinstri en -1 annars.



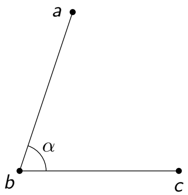
Afstæð hornasumma

- Látum p_i, $i < n$ tákna hornpunkta marghyrnings, q einhvern punkt, $\alpha(a, b, c)$ vera hornið milli a, b og c og $\beta(a, b, c)$ vera 1 ef brotna línustrikið $\langle a, b, c \rangle$ „beygir" til vinstri en -1 annars.
- *Afstæð hornsumma marghyrnings með tilliti til punkts q* er $\sum_{i=0}^{n} \beta(q, p_i, p_{i+1})\alpha(p_i, q, p_{i+1})$.



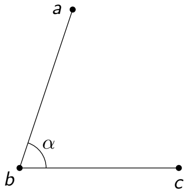
Afstæð hornasumma

- Látum p_i, $i < n$ tákna hornpunkta marghyrnings, q einhvern punkt, $\alpha(a, b, c)$ vera hornið milli a, b og c og $\beta(a, b, c)$ vera 1 ef brotna línustrikið $\langle a, b, c \rangle$ „beygir" til vinstri en -1 annars.
- *Afstæð hornsumma marghyrnings með tilliti til punkts q* er $\sum_{i=0}^{n} \beta(q, p_i, p_{i+1}) \alpha(p_i, q, p_{i+1})$.
- Ef q er inni í marghyrningnum þá er þessi summa bersýnilega 2π.

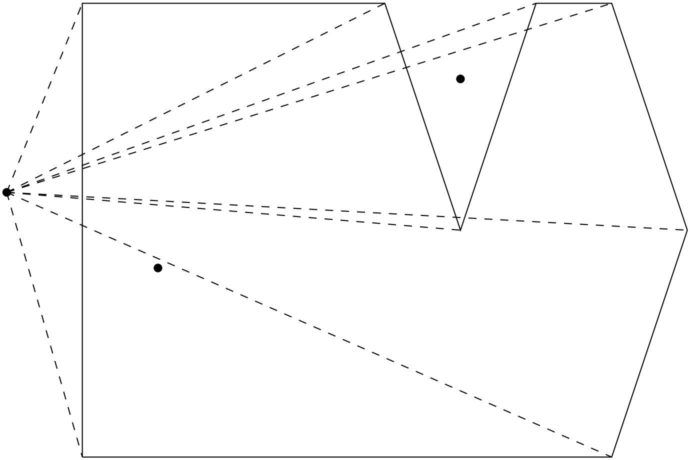


Afstæð hornasumma

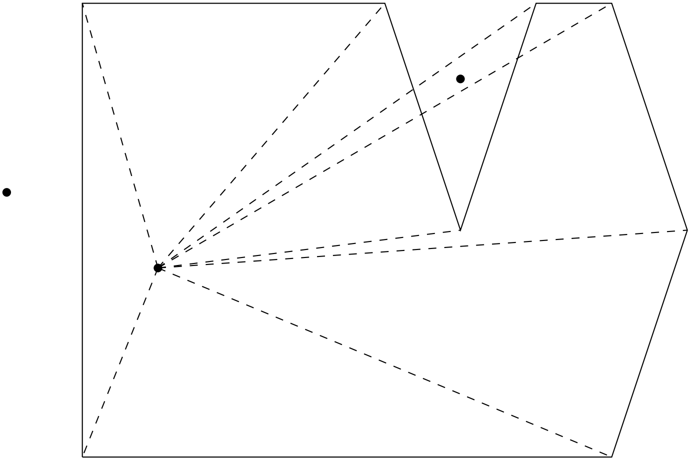
- Látum p_i, $i < n$ tákna hornpunkta marghyrnings, q einhvern punkt, $\alpha(a, b, c)$ vera hornið milli a, b og c og $\beta(a, b, c)$ vera 1 ef brotna línustrikið $\langle a, b, c \rangle$ „beygir" til vinstri en -1 annars.
- *Afstæð hornsumma marghyrnings með tilliti til punkts q* er $\sum_{i=0}^{n} \beta(q, p_i, p_{i+1})\alpha(p_i, q, p_{i+1})$.
- Ef q er inni í marghyrningnum þá er þessi summa bersýnilega 2π.
- Ef q er fyrir utan marghyrninginn þá verður summan hins vegar 0.



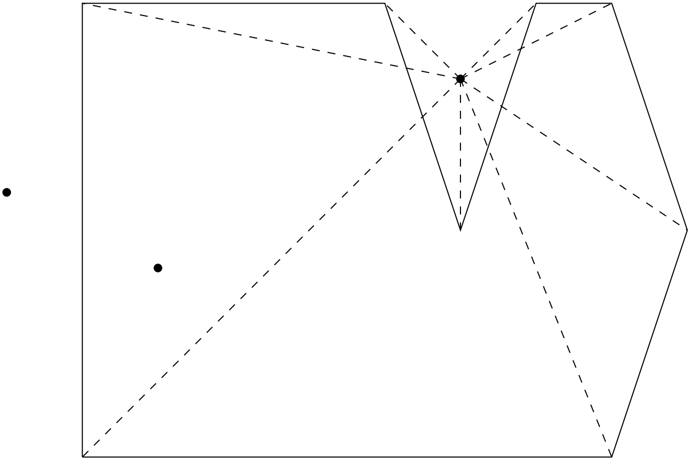
Afstæð hornasumma



Afstæð hornasumma



Afstæð hornasumma



Afstæð hornasumma

```c
double angle(pt a, pt o, pt b)
{ // alpha i glaerunum.
    double r = fabs(carg(a - o) - carg(b - o));
    return r < M_PI ? r : 2*M_PI - r;
}

int ccw(pt a, pt b, pt c)
{ // beta i glaerunum.
    if (cabs(a - b) < EPS || fabs(cimag((c - a)/(b - a))) < EPS)
        return 0;
    return cimag((c - a)/(b - a)) > 0.0 ? 1 : -1;
}

int is_in(pt* p, pt q, int n)
{
    int i;
    double s = 0.0;
    rep(i, n - 1) s += ccw(q, p[i], p[i + 1])*angle(p[i], q, p[i + 1]);
    return (fabs(s) > M_PI ? 1 : 0);
}
```

Kútpur hjúpur punktasafns

- *Kúptur hjúpur punktasafns* er minnsti kúpti marghyrningur sem inniheldur all punkta punktasafnsins.

Kútpur hjúpur punktasafns

- *Kúptur hjúpur punktasafns* er minnsti kúpti marghyrningur sem inniheldur all punkta punktasafnsins.
- Maður getur ímyndað sér að maður taki teygju og strekki hana yfir punkta safnið og sleppi henni svo.

Kútpur hjúpur punktasafns

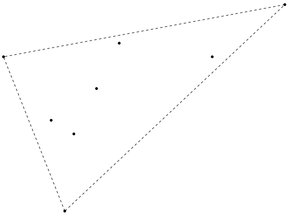
- *Kúptur hjúpur punktasafns* er minnsti kúpti marghyrningur sem inniheldur all punkta punktasafnsins.
- Maður getur ímyndað sér að maður taki teygju og strekki hana yfir punkta safnið og sleppi henni svo.
- Við munum nota aðferð sem kallast *Graham's scan* til að finna kúpta hjúp punktasafns.

Kútpur hjúpur punktasafns

- *Kúptur hjúpur punktasafns* er minnsti kúpti marghyrningur sem inniheldur all punkta punktasafnsins.
- Maður getur ímyndað sér að maður taki teygju og strekki hana yfir punkta safnið og sleppi henni svo.
- Við munum nota aðferð sem kallast *Graham's scan* til að finna kúpta hjúp punktasafns.

Kútpur hjúpur punktasafns

- *Kúptur hjúpur punktasafns* er minnsti kúpti marghyrningur sem inniheldur all punkta punktasafnsins.
- Maður getur ímyndað sér að maður taki teygju og strekki hana yfir punkta safnið og sleppi henni svo.
- Við munum nota aðferð sem kallast *Graham's scan* til að finna kúpta hjúp punktasafns.



Graham's Scan

- Við byrjum á að velja vendipunkt, yfirleitt látinn vera punkturinn neðst til vinstri.

Graham's Scan

- Við byrjum á að velja vendipunkt, yfirleitt látinn vera punkturinn neðst til vinstri.
- Við látum hann fremst í safnið og röðum svo restin miðið við hornið sem þeir mynda við vendipunktinn.

Graham's Scan

- Við byrjum á að velja vendipunkt, yfirleitt látinn vera punkturinn neðst til vinstri.
- Við látum hann fremst í safnið og röðum svo restin miðið við hornið sem þeir mynda við vendipunktinn.
- Við gefum okkur svo hlaða og látum aftasta, fremsta og næst fremsta punktinn á hlaðann.

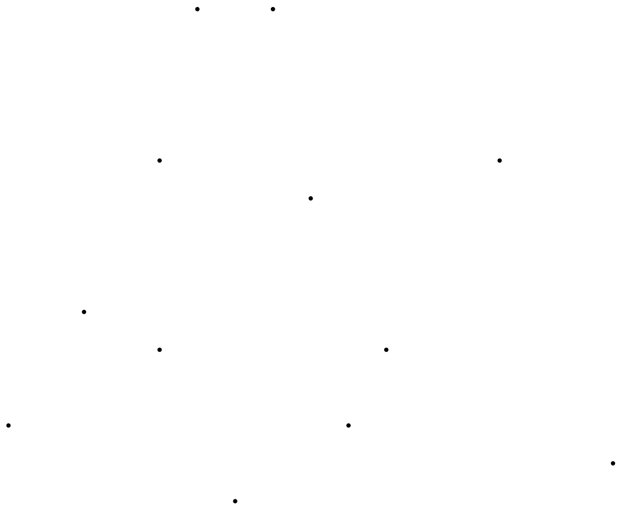
Graham's Scan

- Við byrjum á að velja vendipunkt, yfirleitt látinn vera punkturinn neðst til vinstri.
- Við látum hann fremst í safnið og röðum svo restin miðið við hornið sem þeir mynda við vendipunktinn.
- Við gefum okkur svo hlaða og látum aftasta, fremsta og næst fremsta punktinn á hlaðann.
- Við göngum síðan í gegnum raðaða punkta safnið okkur og fyrir hvert stak fjarlægjum við ofan af hlaðanum á meðan efstu tvö stökin á hlaðanum og stakið sem við erum á í listanum mynda hægri beygju. Þegar þau mynda vinstri beygju bætum við stakinu úr safninu á hlaðan.

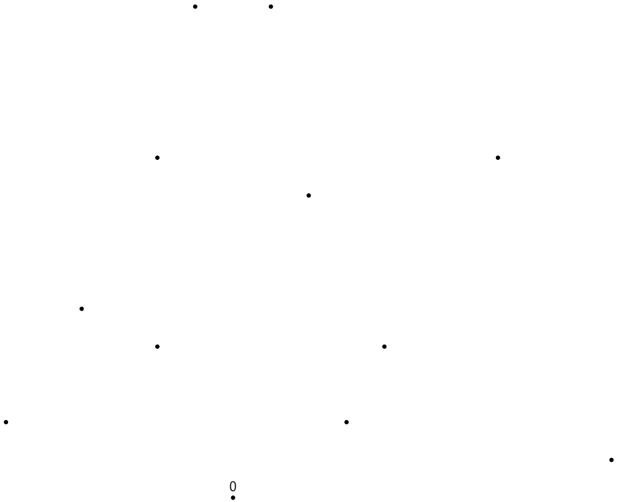
Graham's Scan

- Við byrjum á að velja vendipunkt, yfirleitt látinn vera punkturinn neðst til vinstri.
- Við látum hann fremst í safnið og röðum svo restin miðið við hornið sem þeir mynda við vendipunktinn.
- Við gefum okkur svo hlaða og látum aftasta, fremsta og næst fremsta punktinn á hlaðann.
- Við göngum síðan í gegnum raðaða punkta safnið okkur og fyrir hvert stak fjarlægjum við ofan af hlaðanum á meðan efstu tvö stökin á hlaðanum og stakið sem við erum á í listanum mynda hægri beygju. Þegar þau mynda vinstri beygju bætum við stakinu úr safninu á hlaðan.
- Þegar við erum búin að fara í gegnum allt safnið er hlaðinn kúpti hjúpurinn.

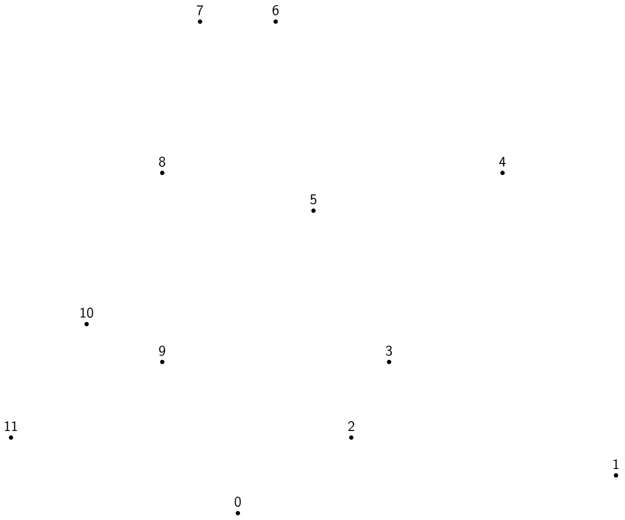
Kútpur hjúpur punktasafns



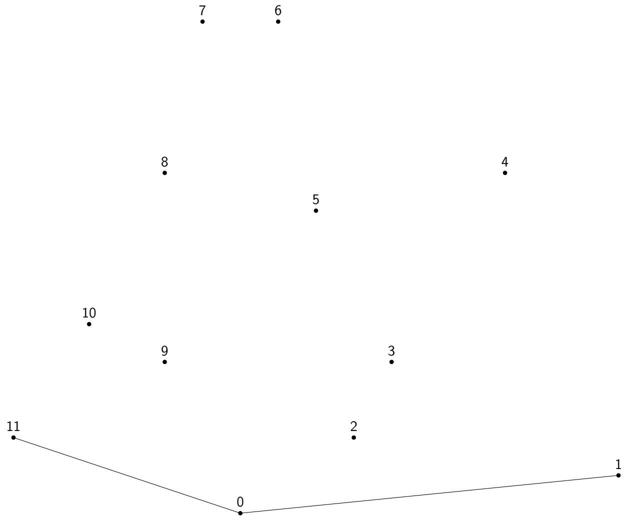
Kútpur hjúpur punktasafns



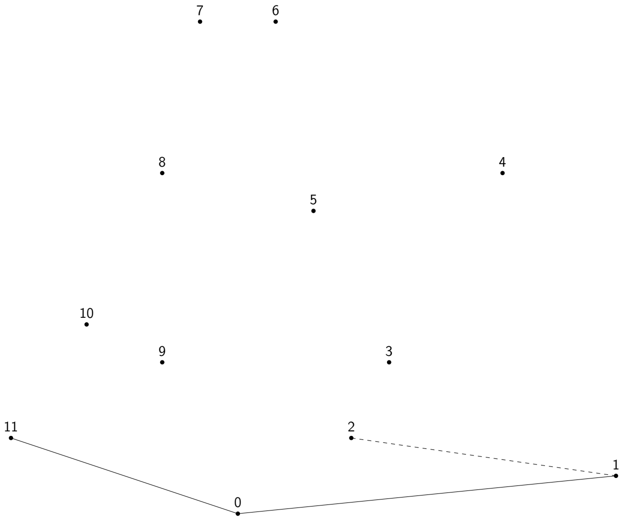
Kútpur hjúpur punktasafns



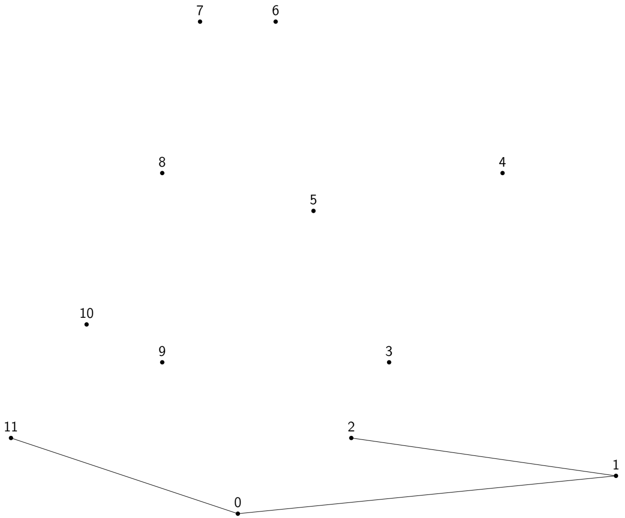
Kútpur hjúpur punktasafns



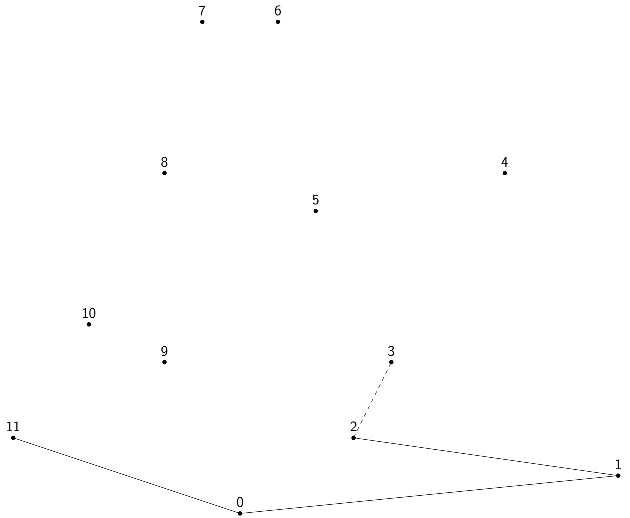
Kútpur hjúpur punktasafns



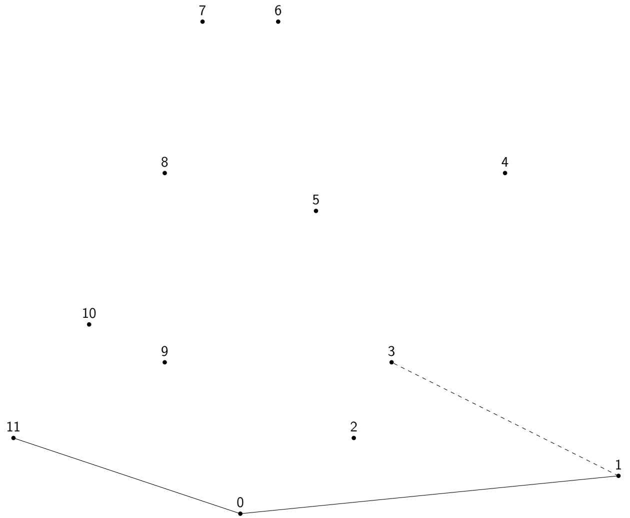
Kútpur hjúpur punktasafns



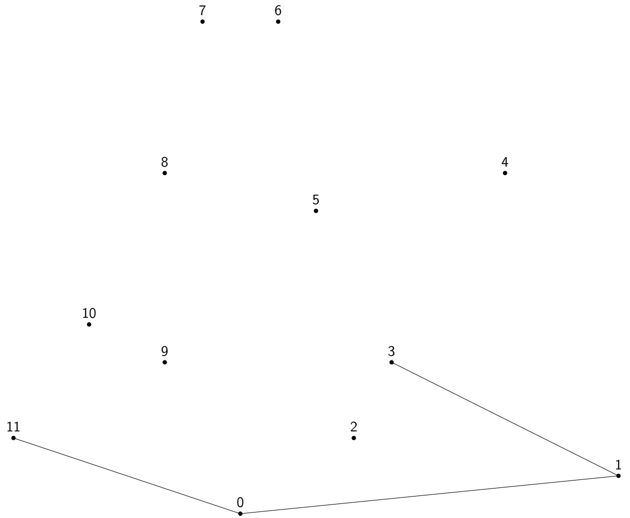
Kútpur hjúpur punktasafns



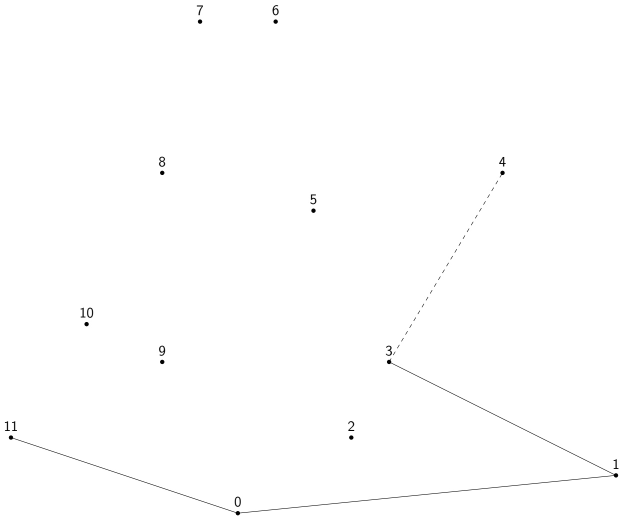
Kútpur hjúpur punktasafns



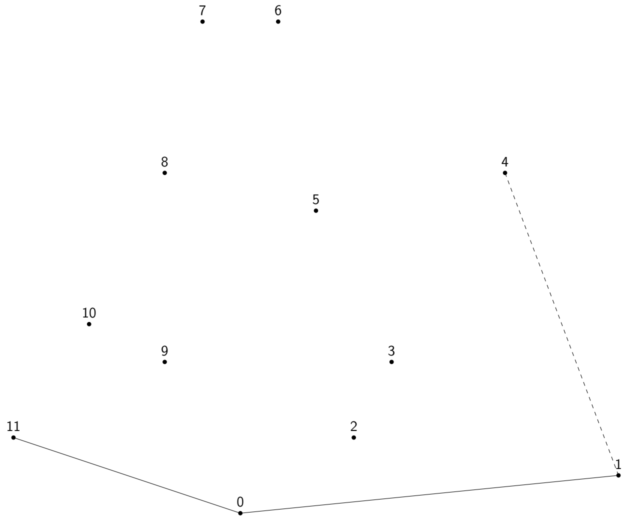
Kútpur hjúpur punktasafns



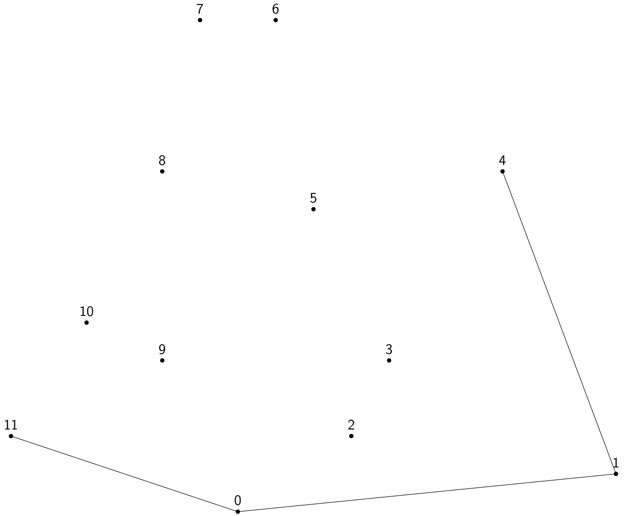
Kútpur hjúpur punktasafns



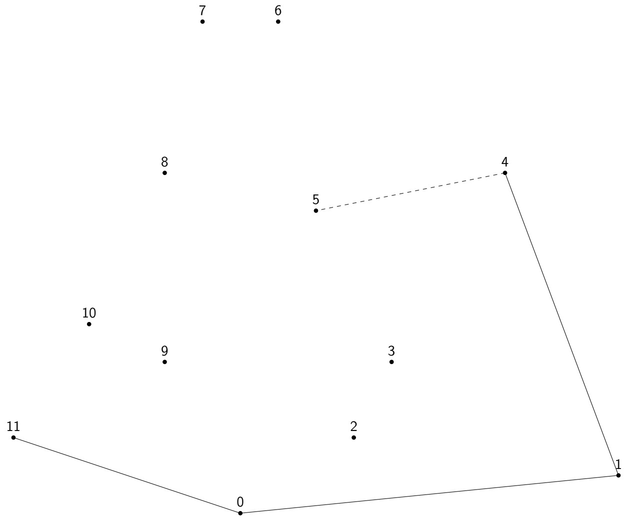
Kútpur hjúpur punktasafns



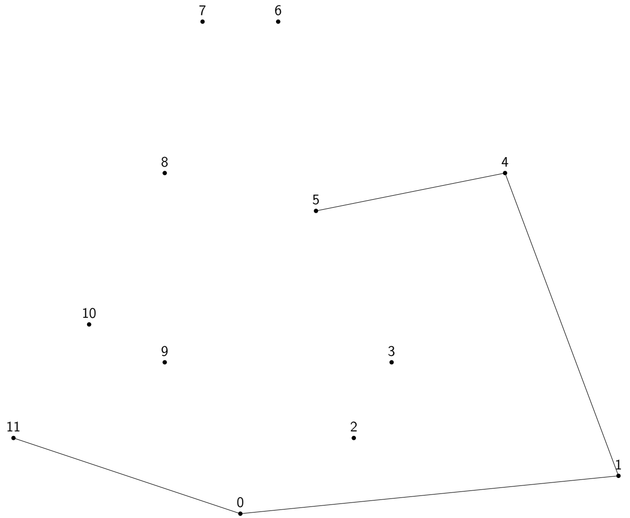
Kútpur hjúpur punktasafns



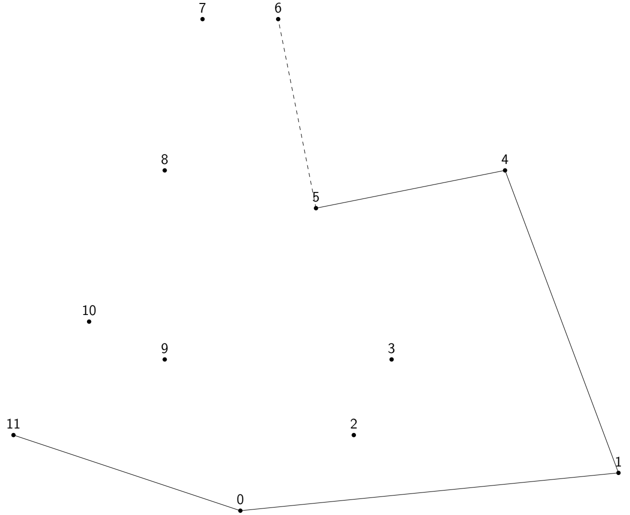
Kútpur hjúpur punktasafns



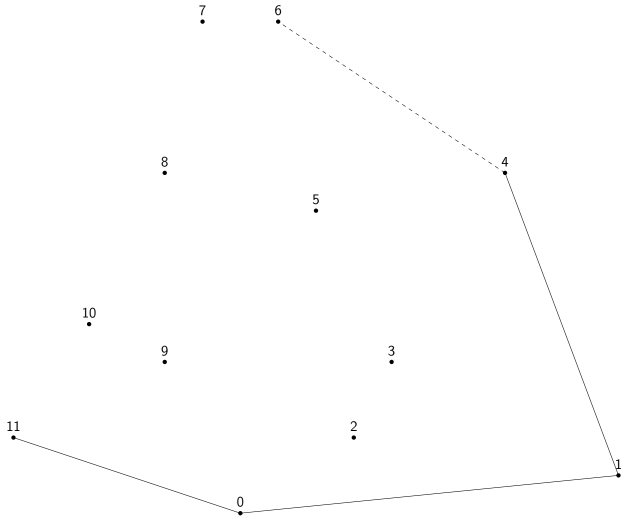
Kútpur hjúpur punktasafns



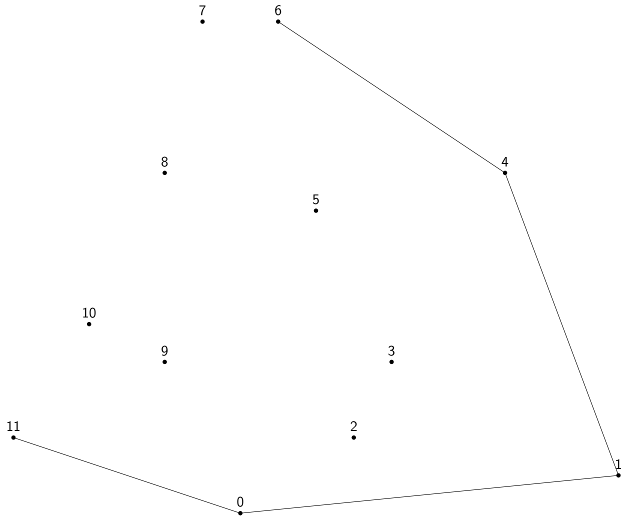
Kútpur hjúpur punktasafns



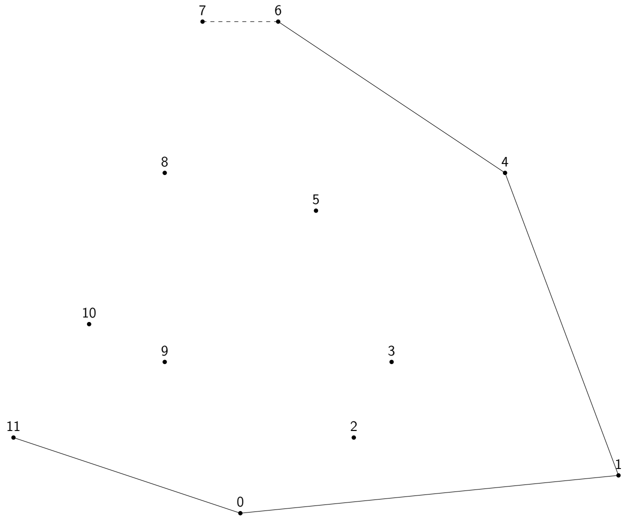
Kútpur hjúpur punktasafns



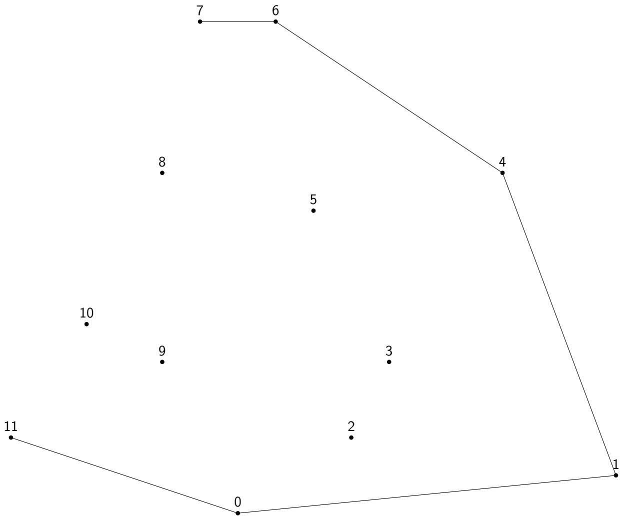
Kútpur hjúpur punktasafns



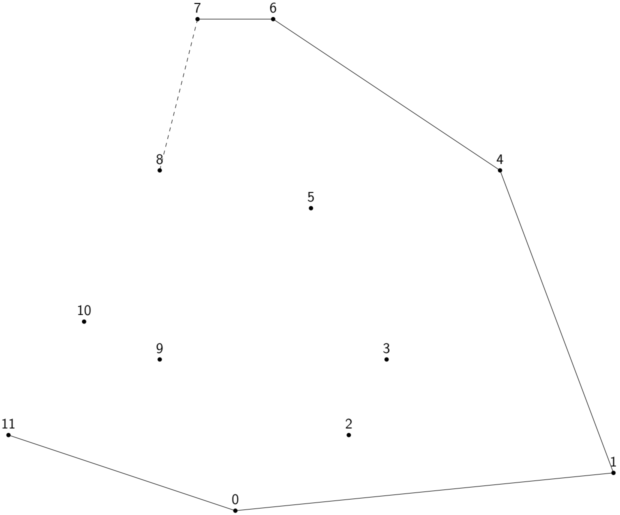
Kútpur hjúpur punktasafns



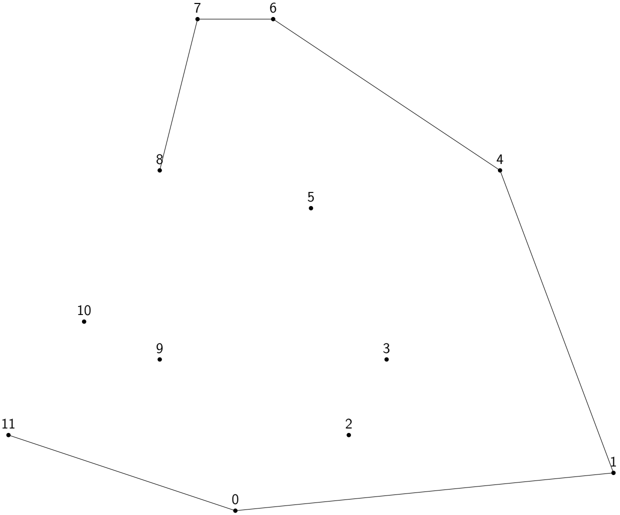
Kútpur hjúpur punktasafns



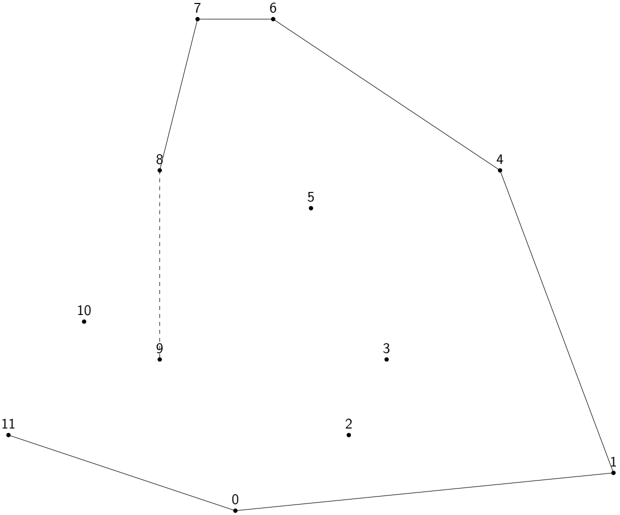
Kútpur hjúpur punktasafns



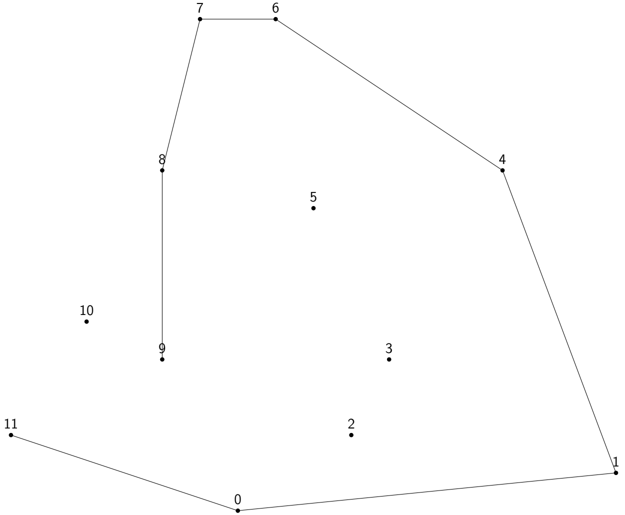
Kútpur hjúpur punktasafns



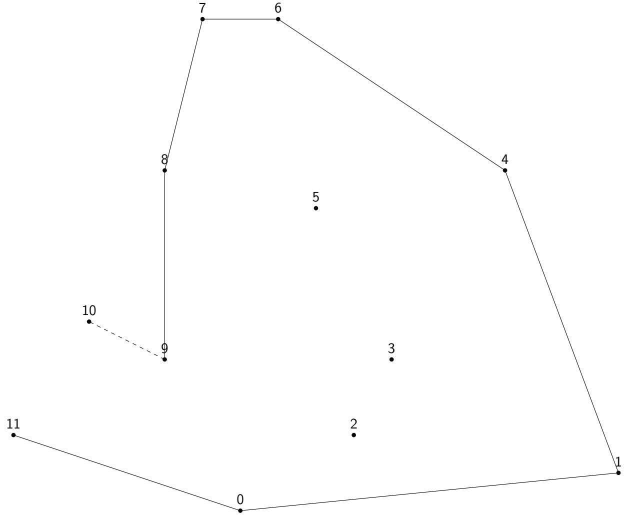
Kútpur hjúpur punktasafns



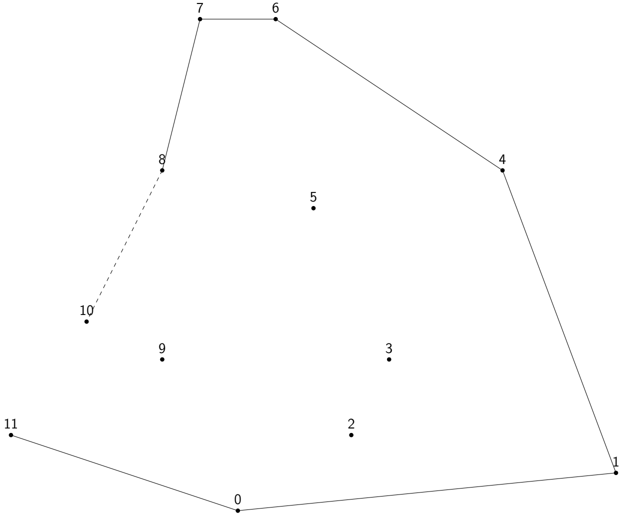
Kútpur hjúpur punktasafns



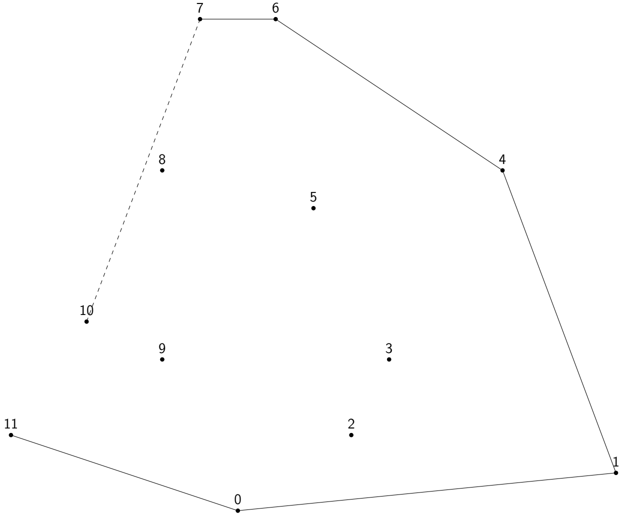
Kútpur hjúpur punktasafns



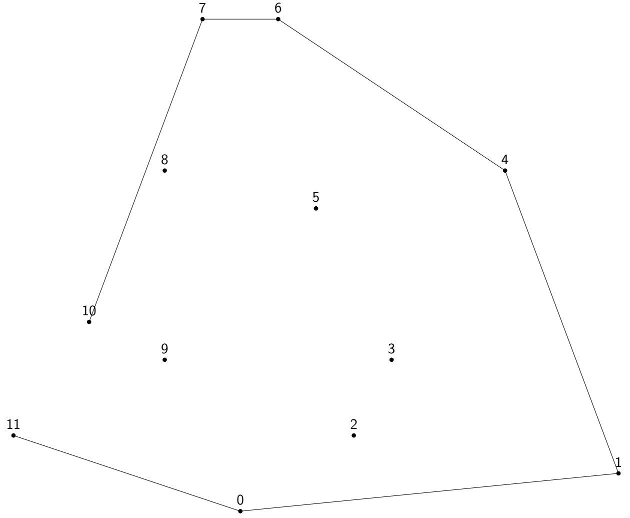
Kútpur hjúpur punktasafns



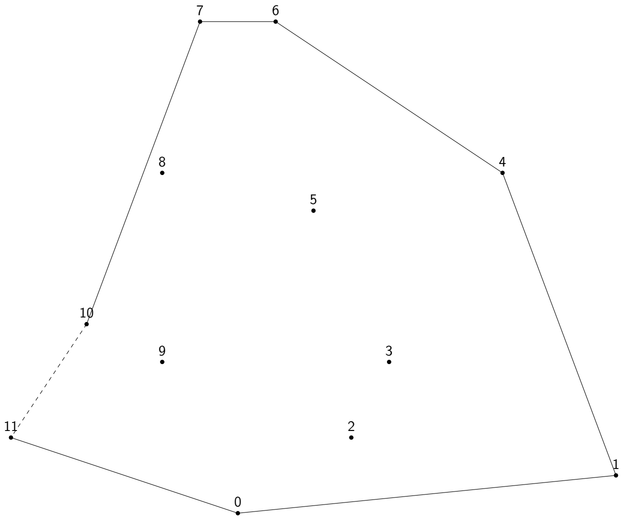
Kútpur hjúpur punktasafns



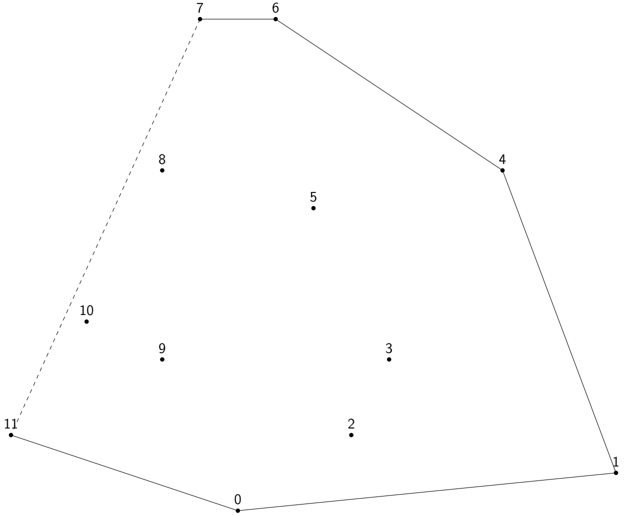
Kútpur hjúpur punktasafns



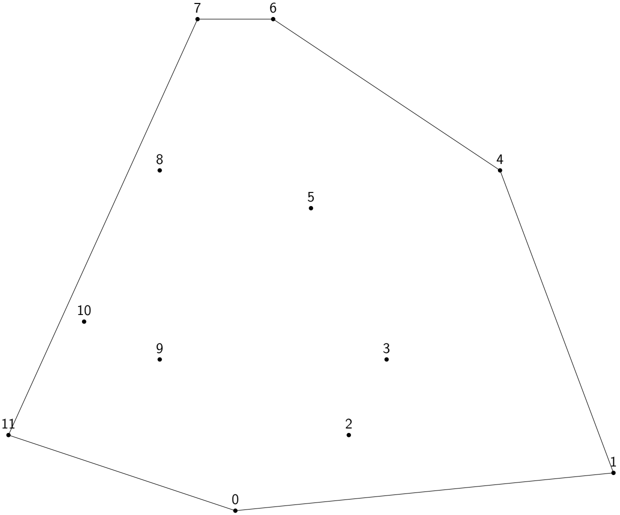
Kútpur hjúpur punktasafns



Kútpur hjúpur punktasafns



Kútpur hjúpur punktasafns



Graham's Scan

- Það er ljóst að í lok reikniritsins lýsir hlaðinn kúptum marghyrningi.

Graham's Scan

- Það er ljóst að í lok reikniritsins lýsir hlaðinn kúptum marghyrningi.
- Það er þó aðeins meira mál að sýna að þetta sé í raun kúpti hjúpur punktasafnsins.

Graham's Scan

- Það er ljóst að í lok reikniritsins lýsir hlaðinn kúptum marghyrningi.
- Það er þó aðeins meira mál að sýna að þetta sé í raun kúpti hjúpur punktasafnsins.
- Við látum það ógert í þessum fyrirlestri.

Graham's Scan

- Það er ljóst að í lok reikniritsins lýsir hlaðinn kúptum marghyrningi.
- Það er þó aðeins meira mál að sýna að þetta sé í raun kúpti hjúpur punktasafnsins.
- Við látum það ógert í þessum fyrirlestri.
- Ef punktasafnið inniheldur n punkta þá er reikniritið $\mathcal{O}(n \log n)$ út af því við þurfum að raða punktunum. Eftir röðun er reikniritið $\mathcal{O}(n)$ því hver punktur fer inn á hlaðan einu sinni.

Graham's Scan

```
pt piv;
int cmp(const void* p1, const void* p2)
{
    pt a = *(pt*)p1, b = *(pt*)p2;
    if (fabs(carg(a - piv) - carg(b - piv)) > EPS)
        return carg(a - piv) < carg(b - piv) ? -1 : 1;
    if (fabs(cabs(a - piv) - cabs(b - piv)) < EPS) return 0;
        return cabs(a - piv) < cabs(b - piv) ? -1 : 1;
}

int convex_hull(pt* p, pt* h, int n)
{
    int j = 0, i, mn = 0;
    for (i = 1; i < n; i++)
        if (cimag(p[i]) < cimag(p[mn]) || cimag(p[i]) == cimag(p[mn])
                && creal(p[i]) < creal(p[mn])) mn = i;
    pt t = p[mn]; p[mn] = p[0]; p[0] = t;
    piv = p[0];
    qsort(p + 1, n - 1, sizeof(p[1]), cmp);
    for (i = 1; i < n && cabs(p[0] - p[i]) < EPS; i++);
    if (i == n) h[j++] = p[0];
    else if (i == n - 1) h[j++] = p[0], h[j++] = p[n - 1];
    if (i >= n - 1) return j;
    h[j++] = p[n - 1], h[j++] = p[0], h[j++] = p[i];
    if (ccw(h[0], h[1], h[2]) == 0)
        return j - (cabs(h[0] - h[1]) < EPS ? 2 : 1);
    for (i++; i < n;)
        (cabs(h[j - 1] - p[i]) > EPS && ccw(h[j - 2], h[j - 1], p[i]) == 1)
            ? (h[j++] = p[i++]) : j--;
    return --j;
}
```

Snúa og hliðra

- Við getum oft nýtt okkur óbreytur í dæmum til að einfalda útrekininga.

Snúa og hliðra

- Við getum oft nýtt okkur óbreytur í dæmum til að einfalda útrekininga.
- Augljósasta dæmið um slíkar óbreytur er að fjarlægð milli tveggja punkta breytist ekki við hliðrun eða snúning.

Snúa og hliðra

- Við getum oft nýtt okkur óbreytur í dæmum til að einfalda útrekininga.
- Augljósasta dæmið um slíkar óbreytur er að fjarlægð milli tveggja punkta breytist ekki við hliðrun eða snúning.
- Við getum nýtt okkur þetta til að finna fjarlægð frá punkti í línu, með

Snúa og hliðra

- Við getum oft nýtt okkur óbreytur í dæmum til að einfalda útrekininga.
- Augljósasta dæmið um slíkar óbreytur er að fjarlægð milli tveggja punkta breytist ekki við hliðrun eða snúning.
- Við getum nýtt okkur þetta til að finna fjarlægð frá punkti í línu, með

```
double p2l(pt p, pt l1, pt l2)
{ // fjarlaegd fra punkti ad linustriki
    p = (p - l1)*cexp(-I*carg(l2 - l1)); l2 -= l1;
    if (-EPS < creal(p) && creal(p) < cabs(l2) + EPS)
        return fabs(cimag(p));
    return fmin(cabs(p), cabs(p - cabs(l2)));
}

double p2l(pt p, pt l1, pt l2)
{ // fjarlaegd fra punkti ad linu
    return fabs(cimag((p - l1)*cexp(-I*carg(l2 - l1))));
}
```

Snúa og hliðra

```
double p2l(pt p, pt l1, pt l2)
{ // fjarlaegd fra punkti ad linustriki
    p = (p - l1)*cexp(-I*carg(l2 - l1)); l2 -= l1;
    if (-EPS < creal(p) && creal(p) < cabs(l2) + EPS)
        return fabs(cimag(p));
    return fmin(cabs(p), cabs(p - cabs(l2)));
}

double p2l(pt p, pt l1, pt l2)
{ // fjarlaegd fra punkti ad linu
    return fabs(cimag((p - l1)*cexp(-I*carg(l2 - l1))));
}
```

Snúa og hliðra

```
double p2l(pt p, pt l1, pt l2)
{ // fjarlaegd fra punkti ad linustriki
    p = (p - l1)*cexp(-I*carg(l2 - l1)); l2 -= l1;
    if (-EPS < creal(p) && creal(p) < cabs(l2) + EPS)
        return fabs(cimag(p));
    return fmin(cabs(p), cabs(p - cabs(l2)));
}

double p2l(pt p, pt l1, pt l2)
{ // fjarlaegd fra punkti ad linu
    return fabs(cimag((p - l1)*cexp(-I*carg(l2 - l1))));
}
```

- Það sem við gerum er að hliðra öllu þannig að línan skeri $(0, 0)$ og snúum svo þannig að línan sé x-ásinn.

Snúa og hliðra

```
double p2l(pt p, pt l1, pt l2)
{ // fjarlaegd fra punkti ad linustriki
    p = (p - l1)*cexp(-I*carg(l2 - l1)); l2 -= l1;
    if (-EPS < creal(p) && creal(p) < cabs(l2) + EPS)
        return fabs(cimag(p));
    return fmin(cabs(p), cabs(p - cabs(l2)));
}

double p2l(pt p, pt l1, pt l2)
{ // fjarlaegd fra punkti ad linu
    return fabs(cimag((p - l1)*cexp(-I*carg(l2 - l1))));
}
```

- Það sem við gerum er að hliðra öllu þannig að línan skeri $(0, 0)$ og snúum svo þannig að línan sé x-ásinn.
- Fjarlægð punkts frá x-ásnum er einfaldlega y-hnit punktsins.

Snúa og hliðra

- Þegar maður venst þessu er þetta mjög þægileg leið til að forrita.

Snúa og hliðra

- Þegar maður venst þessu er þetta mjög þægileg leið til að forrita.
- **Gáta:** Hvað gerir, til dæmis, eftirfarandi bútur?

Snúa og hliðra

- Þegar maður venst þessu er þetta mjög þægileg leið til að forrita.
- **Gáta:** Hvað gerir, til dæmis, eftirfarandi bútur?

```
pt foo(pt a, pt b, pt c)
{
  a -= b, c -= b;
  return cconj(a*cexp(-I*carg(c)))*cexp(I*carg(c)) + b;
}
```