Lausnir á rúmfræðidæmi

Bergur Snorrason

30. mars 2022

- Ég mun leysa dæmið *Maximum Number of Colinear Points* með aferð sem nýtist í önnur dæmi.

- Ég mun leysa dæmið *Maximum Number of Colinear Points* með aferð sem nýtist í önnur dæmi.
- Ég leysi það svo aftur með annari aðferð.

Maximum Number of Colinear Points

- Gefnir eru n punktar í plani.

Maximum Number of Colinear Points

- Gefnir eru n punktar í plani.
- Þú vilt velja hlutmengi af þessum punktum þannig að allir punktarnir í hlutmenginu liggi á sömu línunni.

Maximum Number of Colinear Points

- Gefnir eru n punktar í plani.
- Þú vilt velja hlutmengi af þessum punktum þannig að allir punktarnir í hlutmenginu liggi á sömu línunni.
- Hver er stærðin á stærstu hlutmengjunum sem þú getur valið.

- Hvert par af punktum skilgreinir línu.

- Hvert par af punktum skilgreinir línu.
- Við getum því fyrir sérhvert par af punktum ítrað í gegnum alla punktanna og séð hversu margir liggja á línunni.

- Hvert par af punktum skilgreinir línu.
- Við getum því fyrir sérhvert par af punktum ítrað í gegnum alla punktanna og séð hversu margir liggja á línunni.
- Þessi lausn er $\mathcal{O}(n^3)$.

- Hvert par af punktum skilgreinir línu.
- Við getum því fyrir sérhvert par af punktum ítrað í gegnum alla punktanna og séð hversu margir liggja á línunni.
- Þessi lausn er $\mathcal{O}(n^3)$.
- Reynum að bæta þetta.

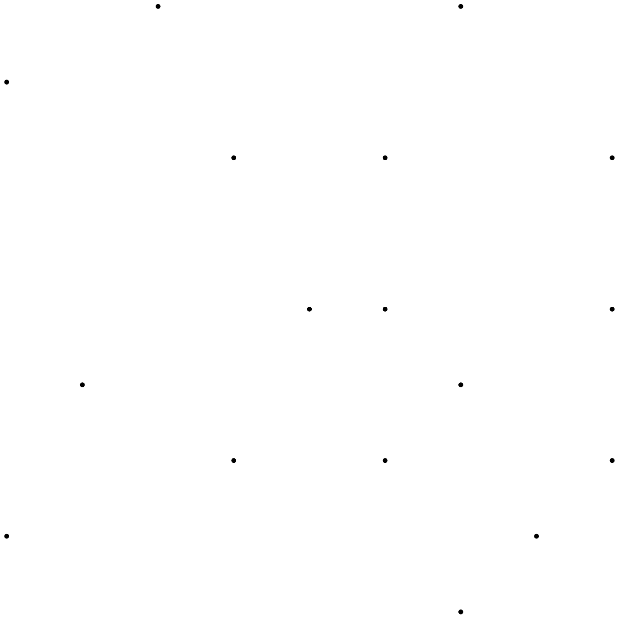
- Veljum einn punkt sem vendipunkt og röðum öðrum punktum miðað við stefnuhornið sem þeir mynda við vendipuktinn.

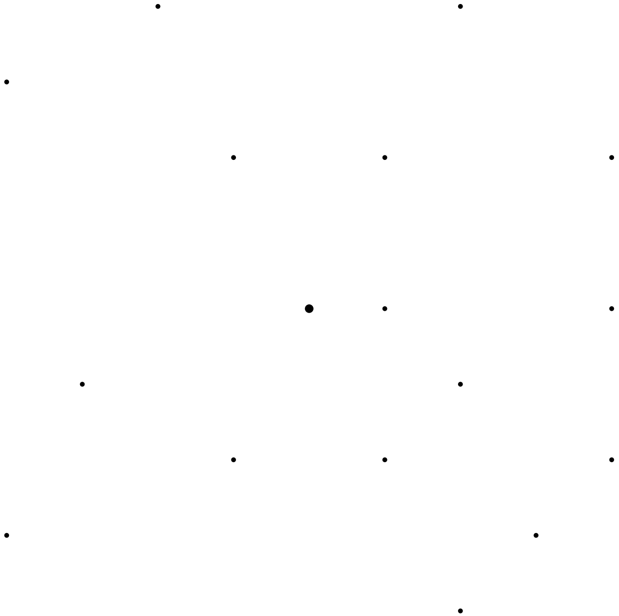
- Veljum einn punkt sem vendipunkt og röðum öðrum punktum miðað við stefnuhornið sem þeir mynda við vendipuktinn.
- Þá eru allir punktar sem liggja á sama geisla sem byrjar í vendipunktinum aðlægir.

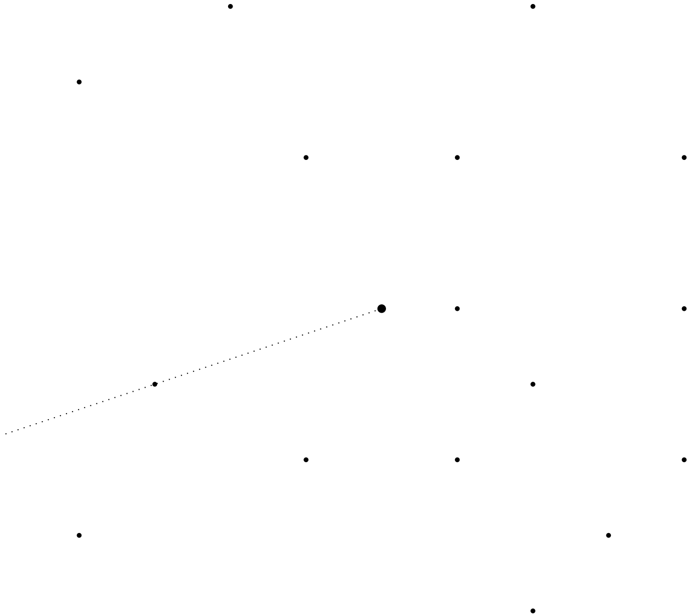
- Veljum einn punkt sem vendipunkt og röðum öðrum punktum miðað við stefnuhornið sem þeir mynda við vendipuktinn.
- Þá eru allir punktar sem liggja á sama geisla sem byrjar í vendipunktinum aðlægir.
- Við getum því gengið einu sinni í gegnum punktanna og fundið besta svarið að því gefnum að vendipunkturinn liggi á línunni.

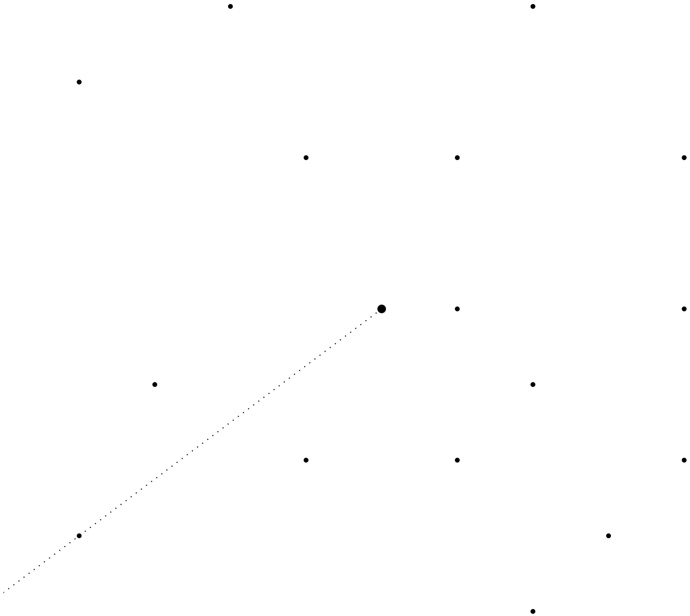
- Veljum einn punkt sem vendipunkt og röðum öðrum punktum miðað við stefnuhornið sem þeir mynda við vendipuktinn.
- Þá eru allir punktar sem liggja á sama geisla sem byrjar í vendipunktinum aðlægir.
- Við getum því gengið einu sinni í gegnum punktanna og fundið besta svarið að því gefnum að vendipunkturinn liggi á línunni.
- Við getum endurtekið þetta n sinnum, þannig að hver punktur fær að vera vendipunktur.

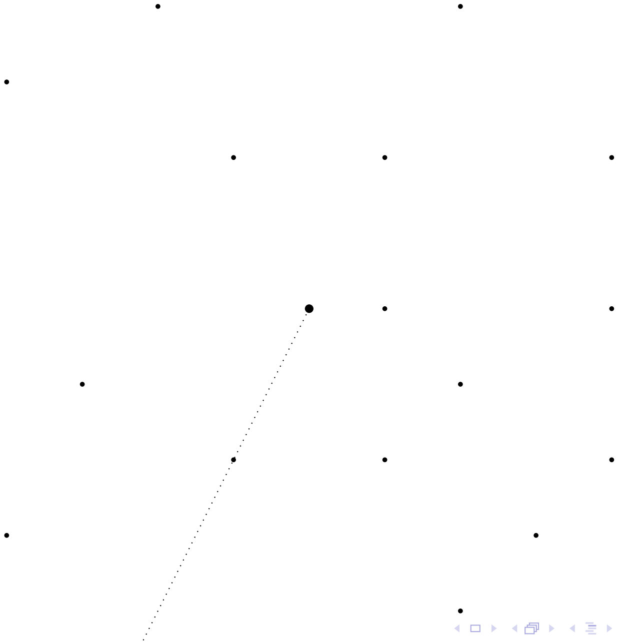
- Veljum einn punkt sem vendipunkt og röðum öðrum punktum miðað við stefnuhornið sem þeir mynda við vendipuktinn.
- Þá eru allir punktar sem liggja á sama geisla sem byrjar í vendipunktinum aðlægir.
- Við getum því gengið einu sinni í gegnum punktanna og fundið besta svarið að því gefnum að vendipunkturinn liggi á línunni.
- Við getum endurtekið þetta n sinnum, þannig að hver punktur fær að vera vendipunktur.
- Tökum sýnidæmi.

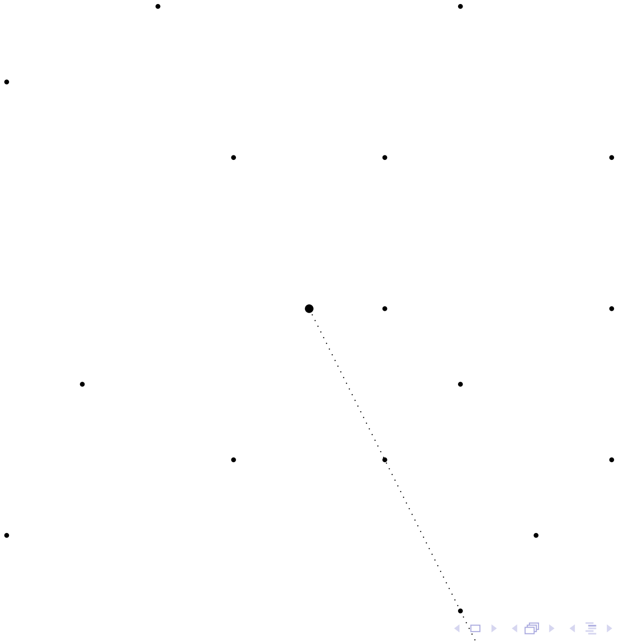


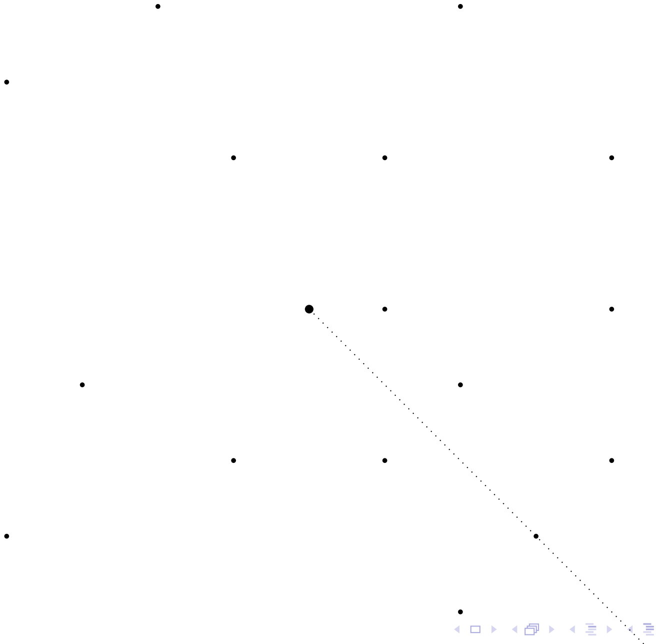


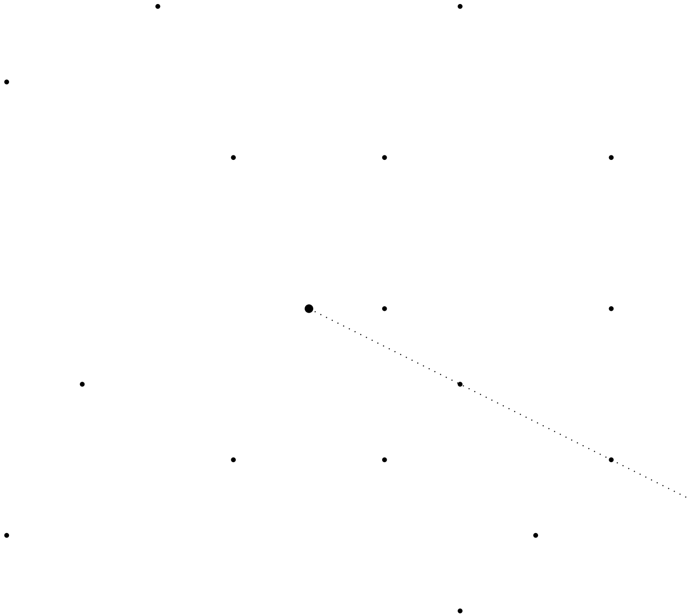


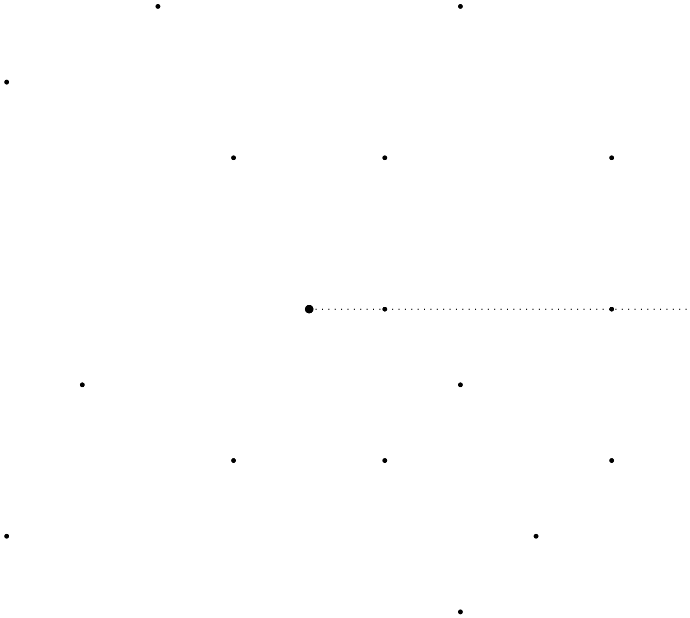


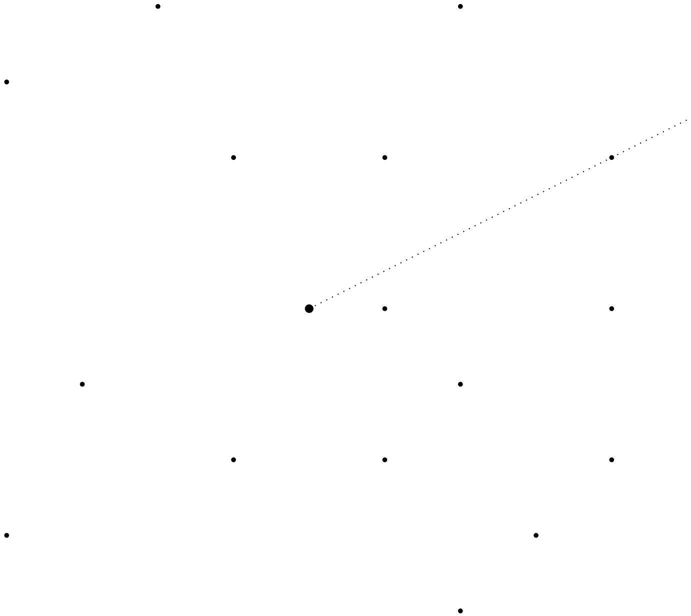


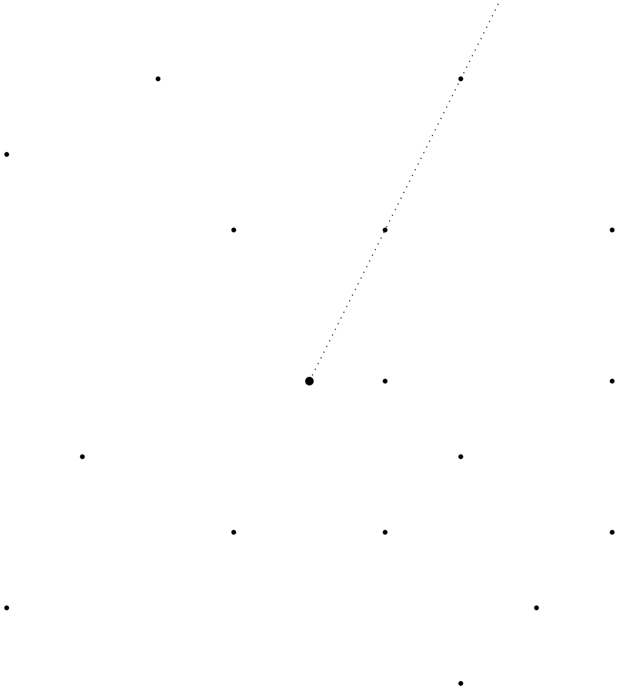


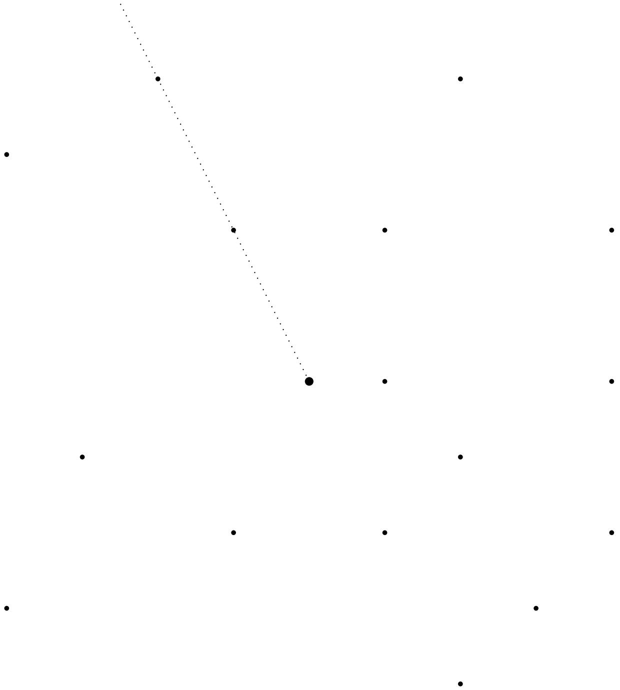


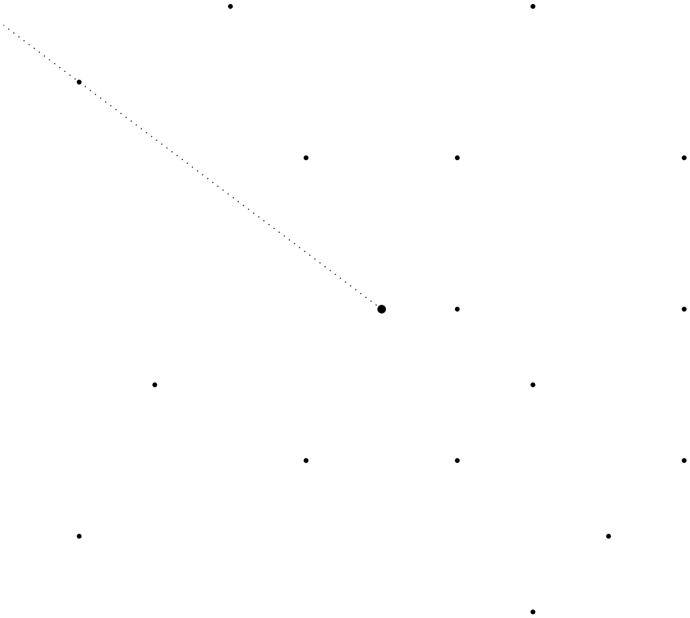


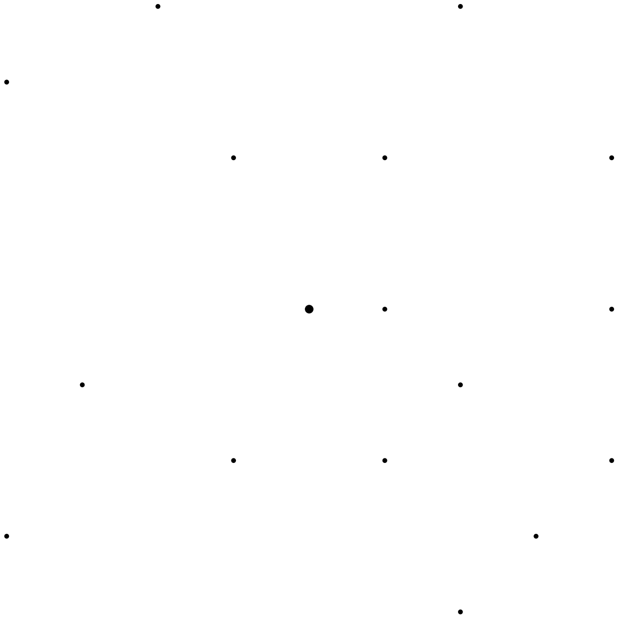


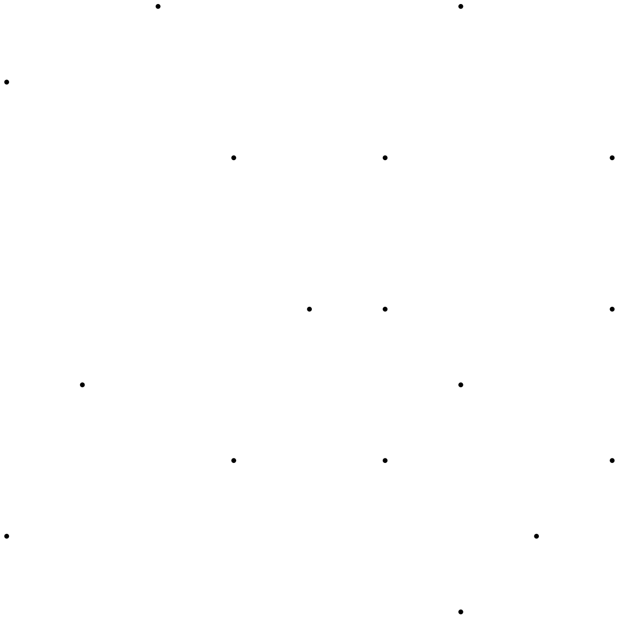


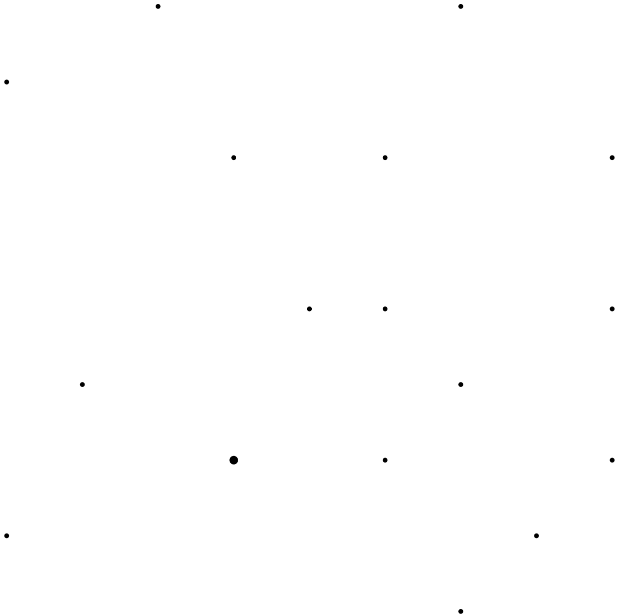


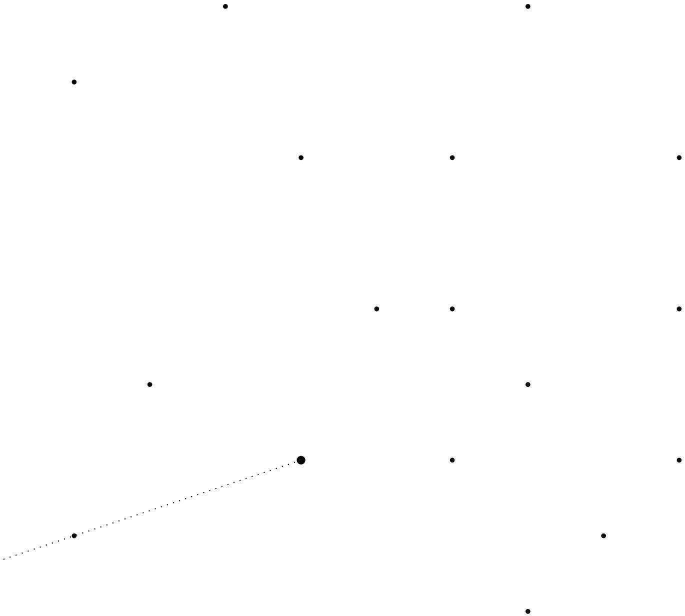


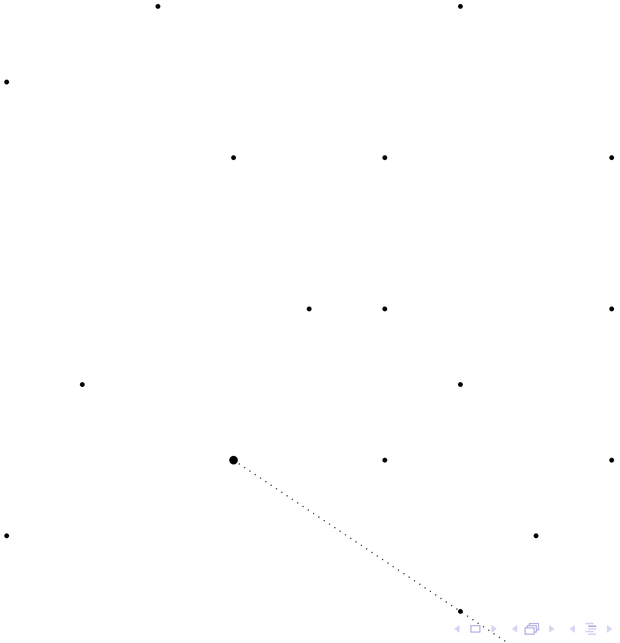


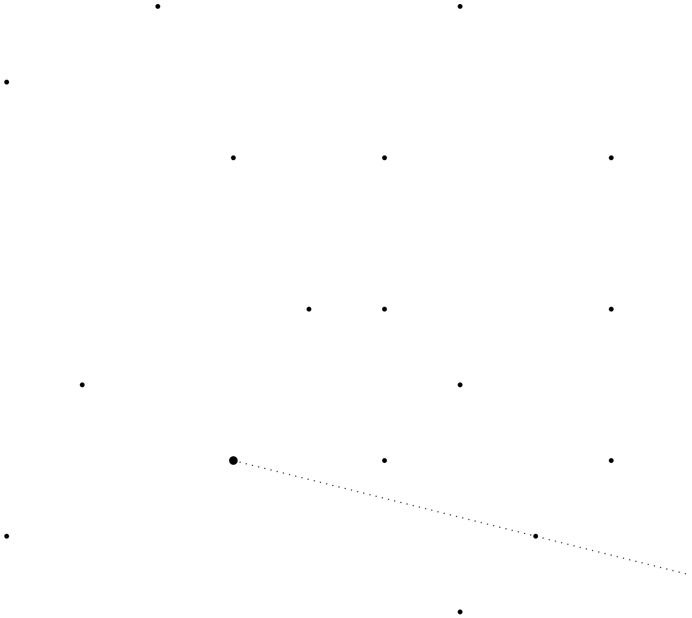


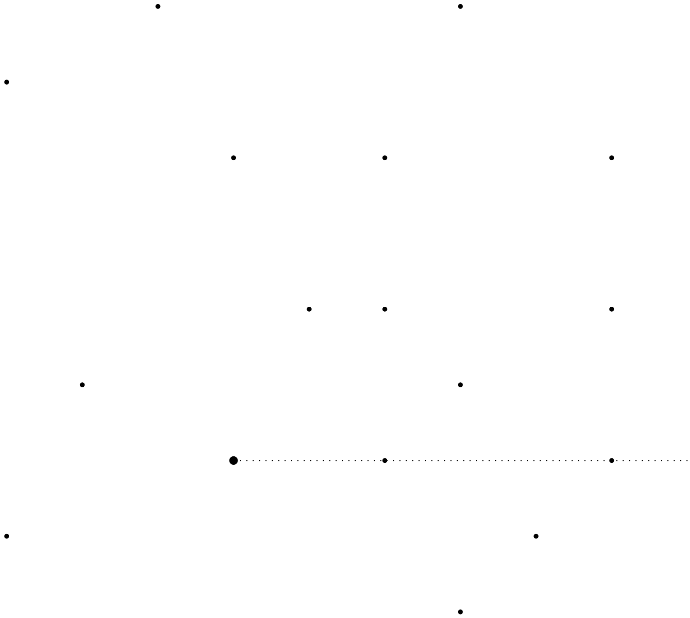


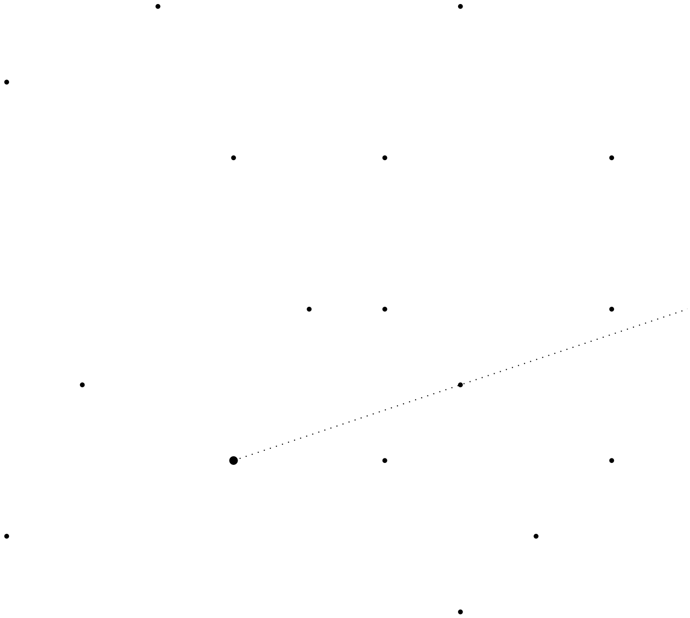


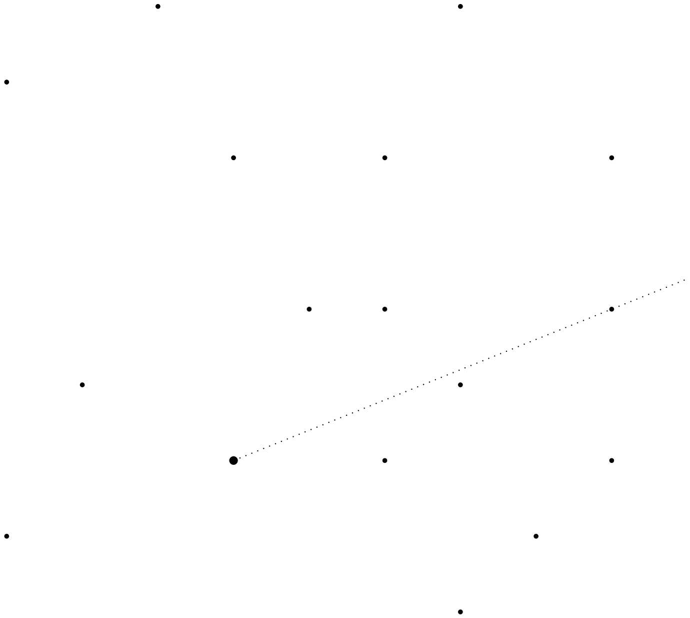


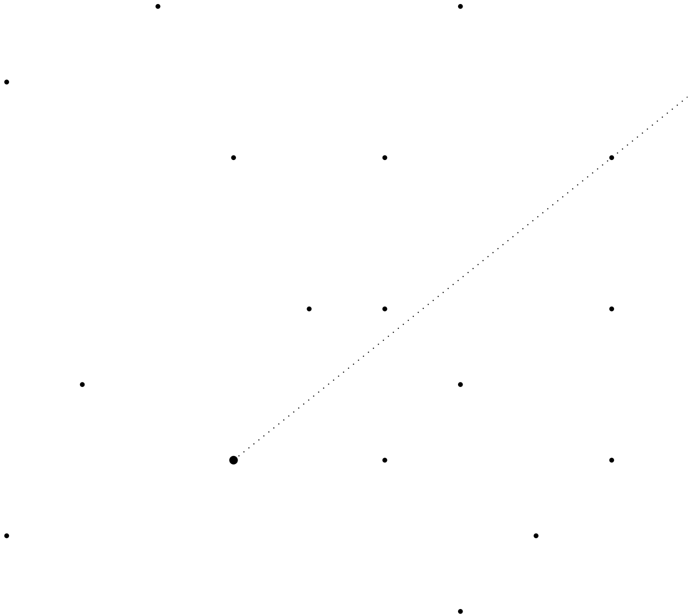


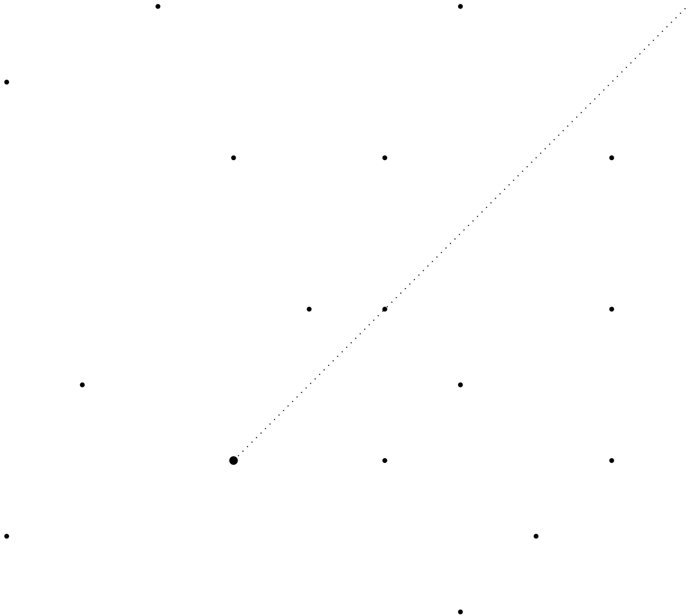


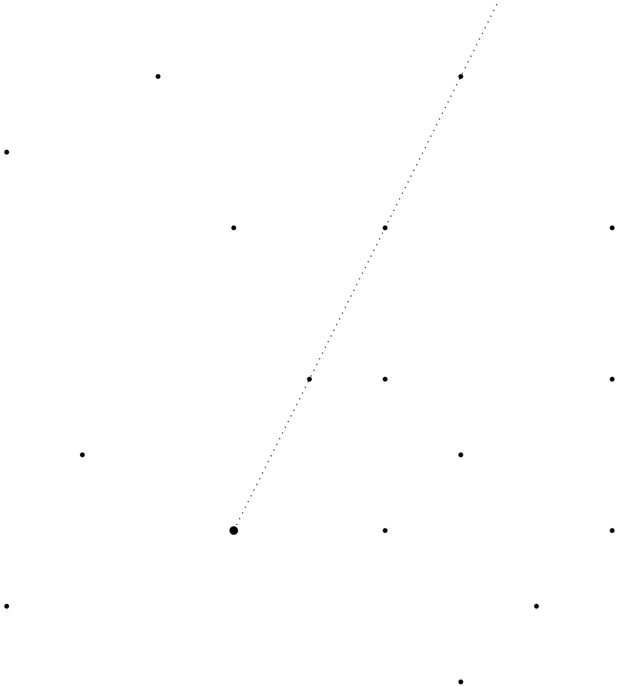


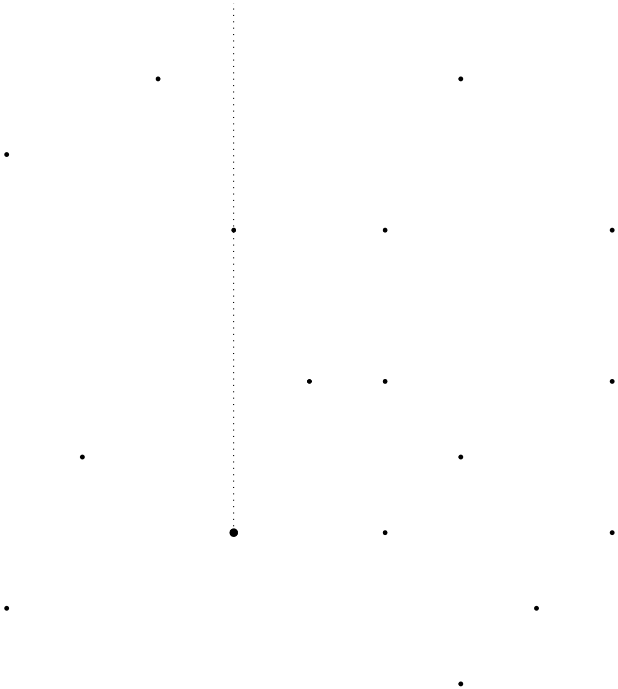


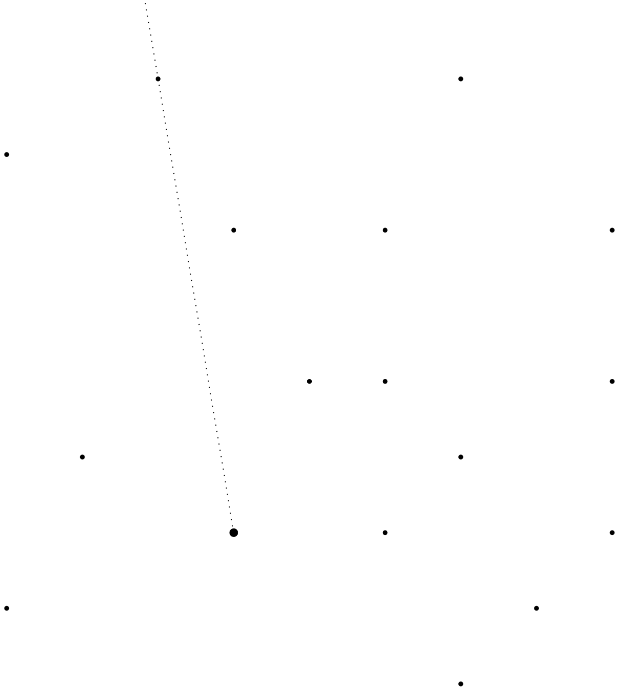


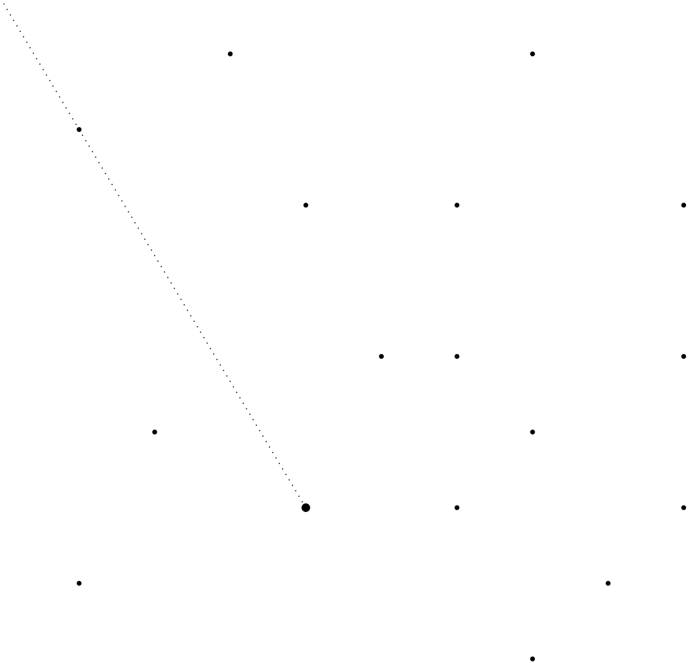


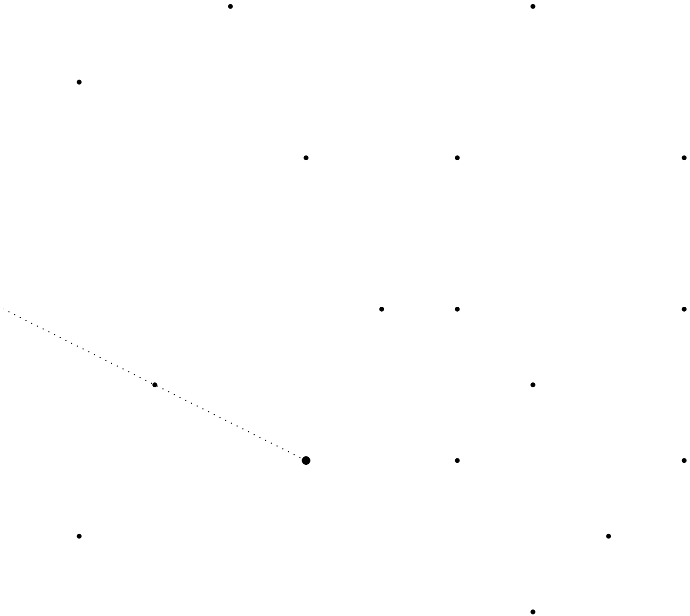


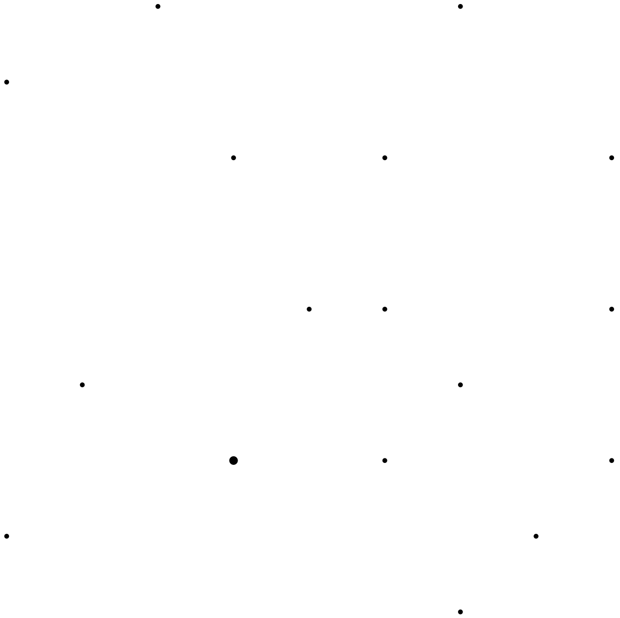


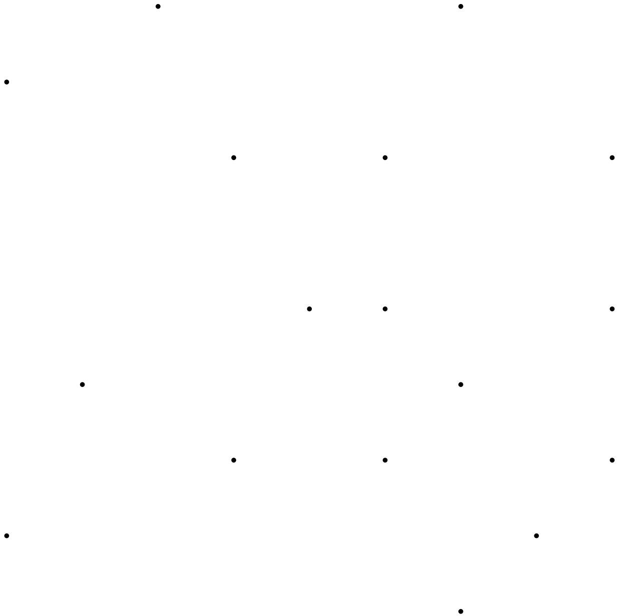


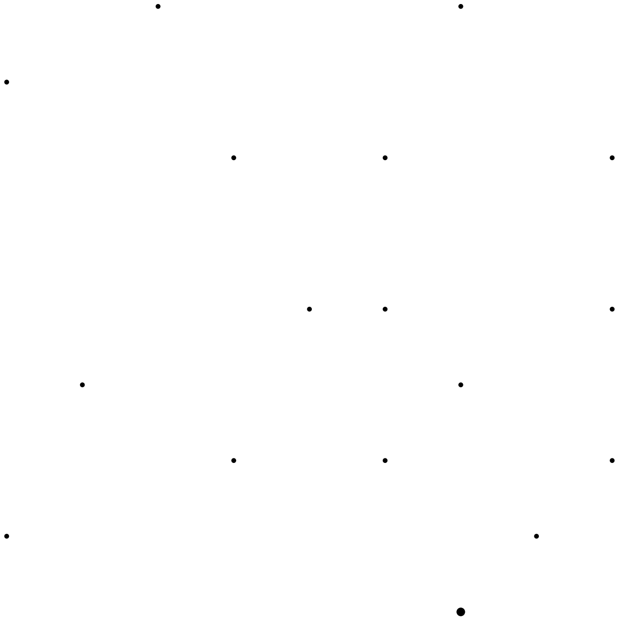


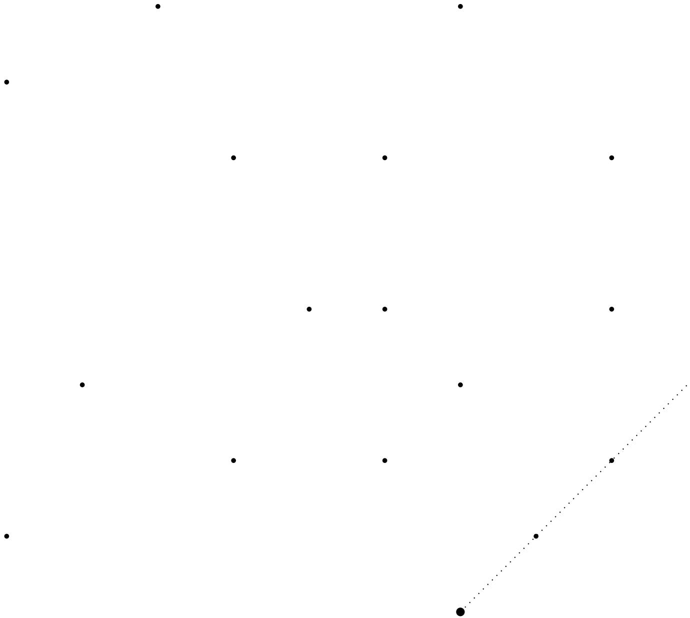


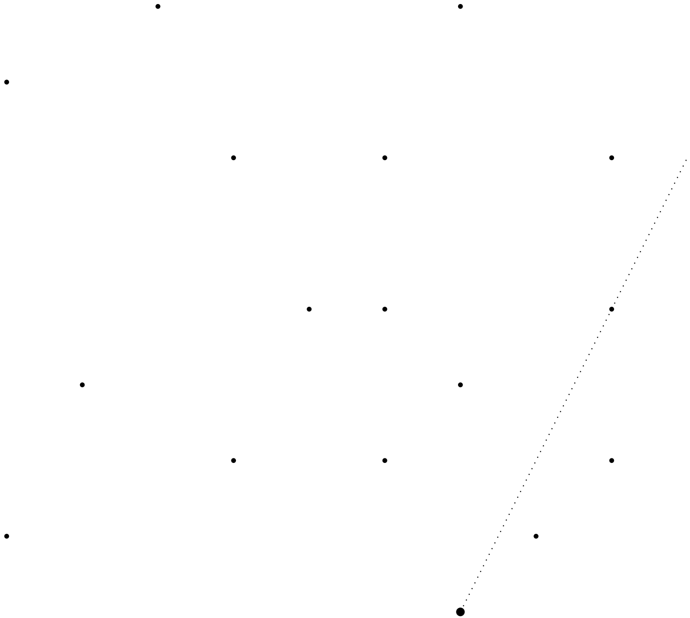


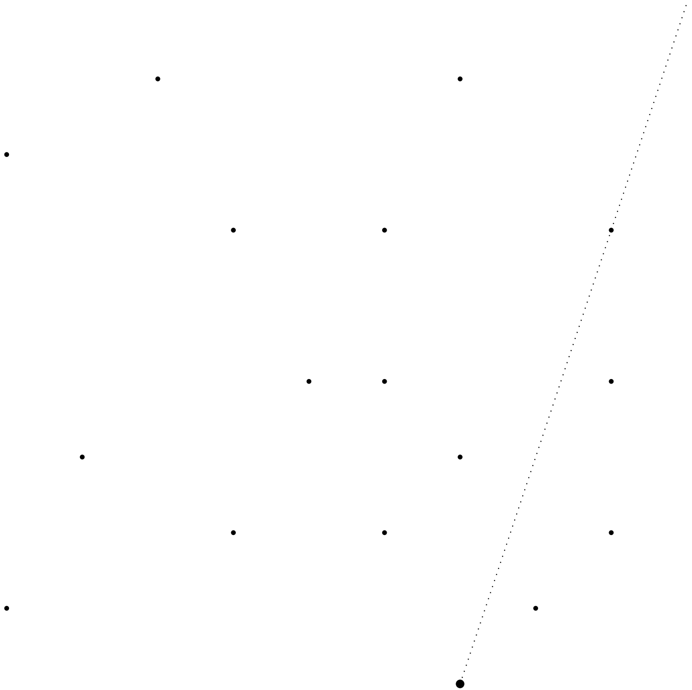


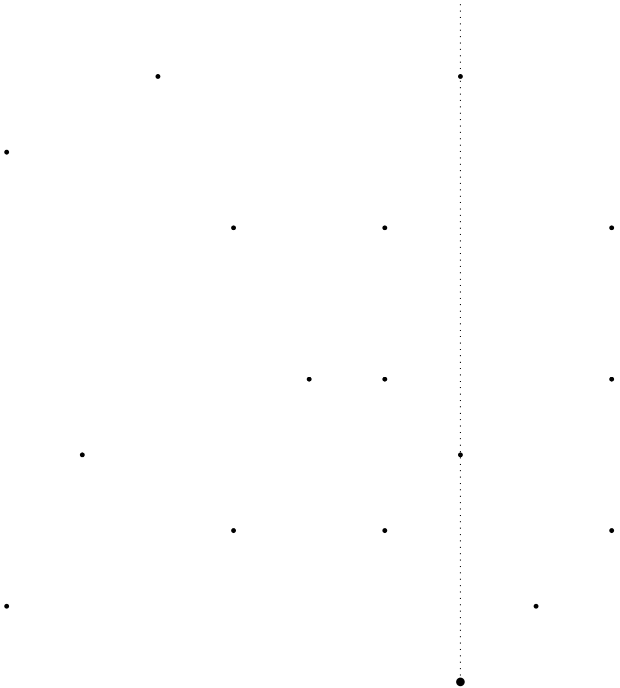


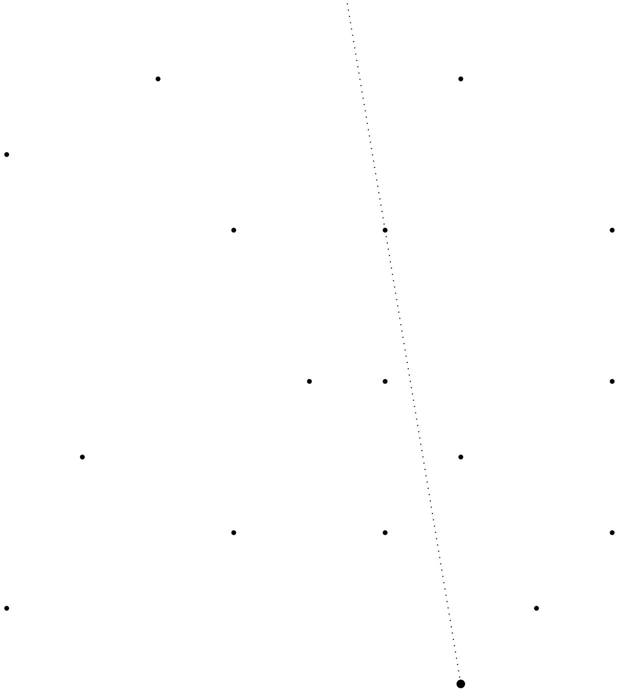


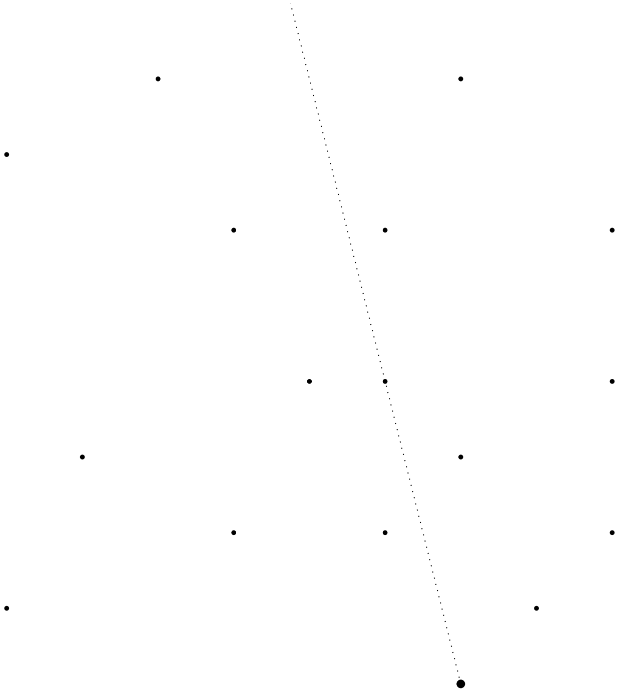


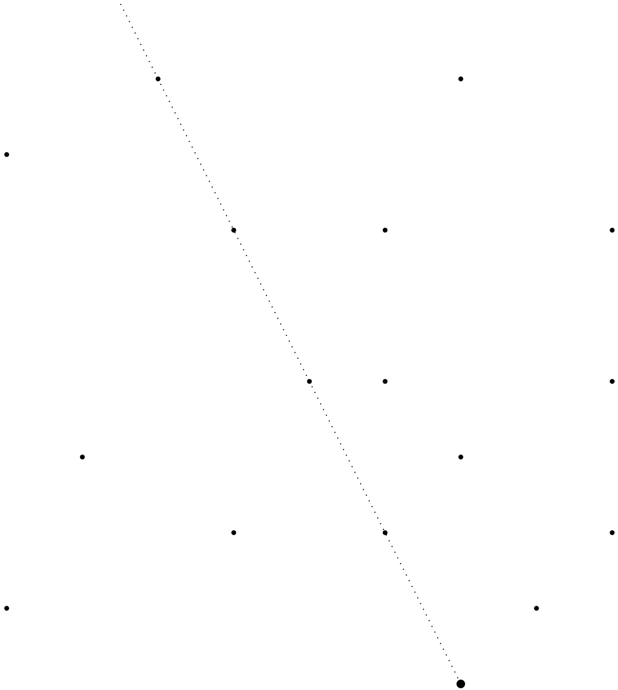


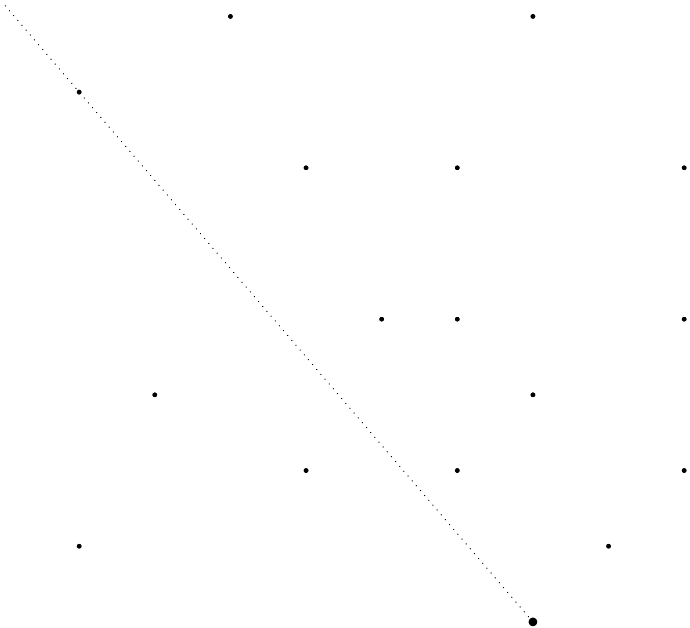


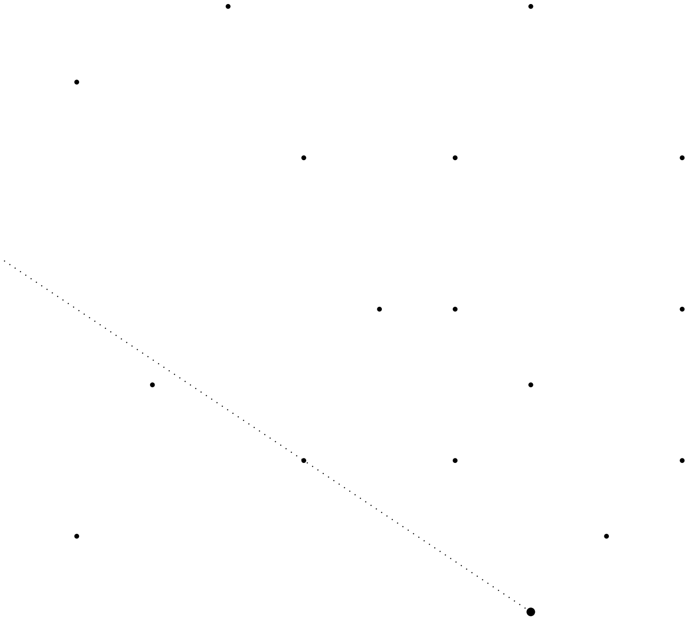


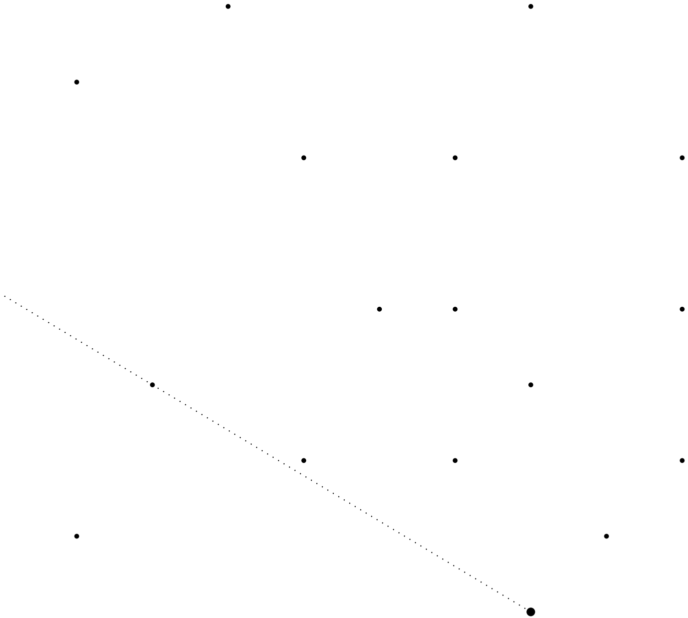


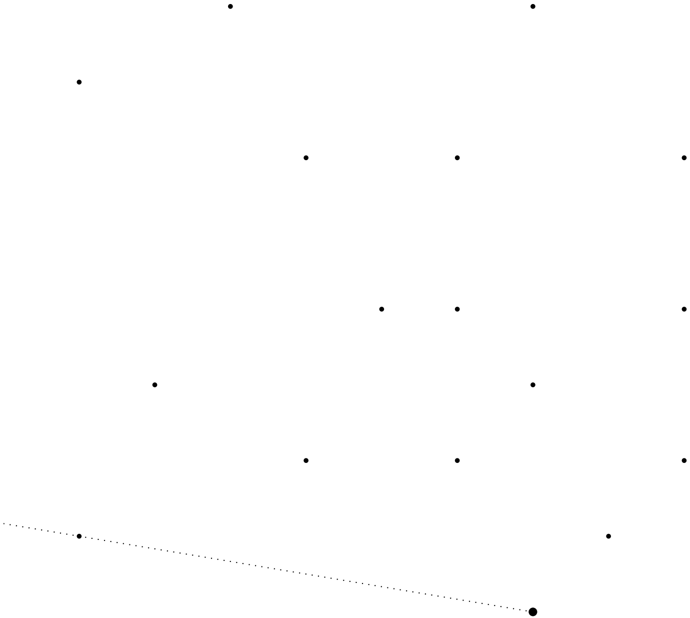


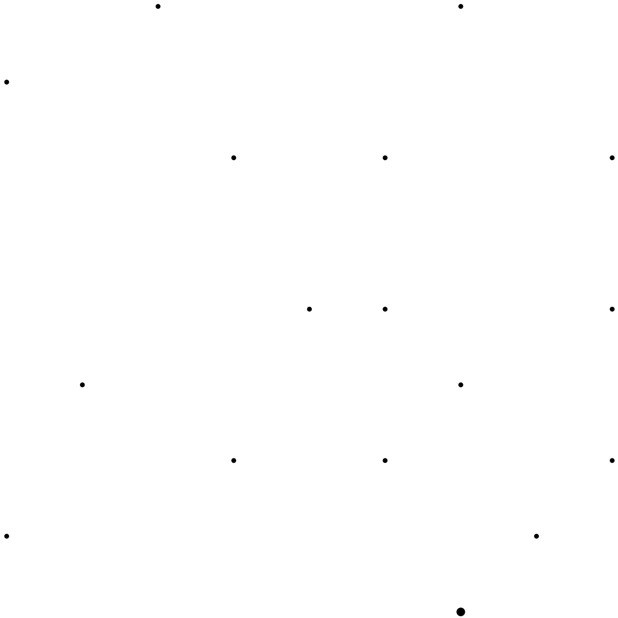


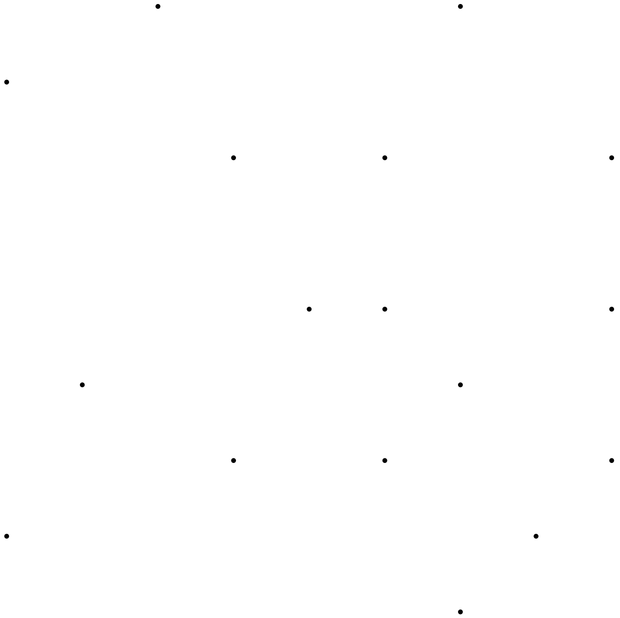












- Við köllum þessa aðferð *sópinn* (e. *sweep line*).

- Við köllum þessa aðferð *sópinn* (e. *sweep line*).
- Dæmi sem má leysa með sópnum:

- Við köllum þessa aðferð *sópinn* (e. *sweep line*).
- Dæmi sem má leysa með sópnum:
  - Finna nálægustu tvo punkta í punktasafni.

- Við köllum þessa aðferð *sópinn* (e. *sweep line*).
- Dæmi sem má leysa með sópnum:
  - Finna nálægustu tvo punkta í punktasafni.
  - Finna Delaunay net punktasafns (reiknirit Fortunes).

- Við köllum þessa aðferð *sópinn* (e. *sweep line*).
- Dæmi sem má leysa með sópnum:
  - Finna nálægustu tvo punkta í punktasafni.
  - Finna Delaunay net punktasafns (reiknirit Fortunes).
  - Athuga hvort safn línustrika skerist (reiknirit Shamosar og Hoeys).

- Við köllum þessa aðferð *sópinn* (e. *sweep line*).
- Dæmi sem má leysa með sópnum:
  - Finna nálægustu tvo punkta í punktasafni.
  - Finna Delaunay net punktasafns (reiknirit Fortunes).
  - Athuga hvort safn línustrika skerist (reiknirit Shamosar og Hoeys).
- Þessi dæmi eiga það öll sameiginlegt að við drögum beina línu (yfirleitt samsíða y-ásnum) í gegnum punktasafnið okkar.

- Við köllum þessa aðferð *sópinn* (e. *sweep line*).
- Dæmi sem má leysa með sópnum:
  - Finna nálægustu tvo punkta í punktasafni.
  - Finna Delaunay net punktasafns (reiknirit Fortunes).
  - Athuga hvort safn línustrika skerist (reiknirit Shamosar og Hoeys).
- Þessi dæmi eiga það öll sameiginlegt að við drögum beina línu (yfirleitt samsíða y-ásnum) í gegnum punktasafnið okkar.
- Dæmið sem við erum að skoða snýr línu með fasta miðju, svo kallaður *snúningssópur*.

- Við köllum þessa aðferð *sópinn* (e. *sweep line*).
- Dæmi sem má leysa með sópnum:
    - Finna nálægustu tvo punkta í punktasafni.
    - Finna Delaunay net punktasafns (reiknirit Fortunes).
    - Athuga hvort safn línustrika skerist (reiknirit Shamosar og Hoeys).
- Þessi dæmi eiga það öll sameiginlegt að við drögum beina línu (yfirleitt samsíða y-ásnum) í gegnum punktasafnið okkar.
- Dæmið sem við erum að skoða snýr línu með fasta miðju, svo kallaður *snúningssópur*.
- Snúningssópurinn er algengari í dæmum.

- Nýtum okkur þetta tækifæri og skoðum hvernig við getum útfært rúmfræði í heiltölum.

- Nýtum okkur þetta tækifæri og skoðum hvernig við getum útfært rúmfræði í heiltölum.
- Helsti kosturinn við þessa aðferð er að við getum leyst dæmið án fleytitöluskekkju.

- Nýtum okkur þetta tækifæri og skoðum hvernig við getum útfært rúmfræði í heiltölum.
- Helsti kosturinn við þessa aðferð er að við getum leyst dæmið án fleytitöluskekkju.
- Við geymum þá punkt sem tvennd af heiltölum.

- Nýtum okkur þetta tækifæri og skoðum hvernig við getum útfært rúmfræði í heiltölum.
- Helsti kosturinn við þessa aðferð er að við getum leyst dæmið án fleytitöluskekkju.
- Við geymum þá punkt sem tvennd af heiltölum.
- Við þurfum að geta raðað punktunum eftir stefnuhorni.

- Nýtum okkur þetta tækifæri og skoðum hvernig við getum útfært rúmfræði í heiltölum.
- Helsti kosturinn við þessa aðferð er að við getum leyst dæmið án fleytitöluskekkju.
- Við geymum þá punkt sem tvennd af heiltölum.
- Við þurfum að geta raðað punktunum eftir stefnuhorni.
- Við getum gert þetta án þess að reikna stefnuhornið.

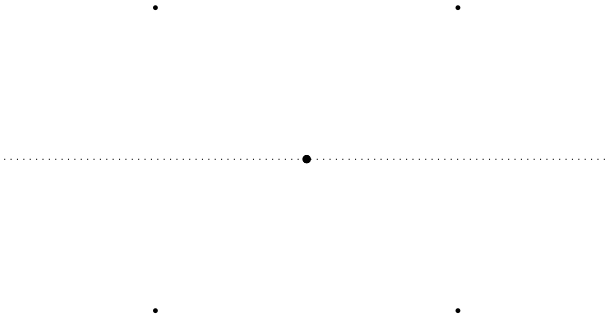
- Nýtum okkur þetta tækifæri og skoðum hvernig við getum útfært rúmfræði í heiltölum.
- Helsti kosturinn við þessa aðferð er að við getum leyst dæmið án fleytitöluskekkju.
- Við geymum þá punkt sem tvennd af heiltölum.
- Við þurfum að geta raðað punktunum eftir stefnuhorni.
- Við getum gert þetta án þess að reikna stefnuhornið.
- Köllum vendipunktinn P og punktanna sem við viljum bera saman X og Y.

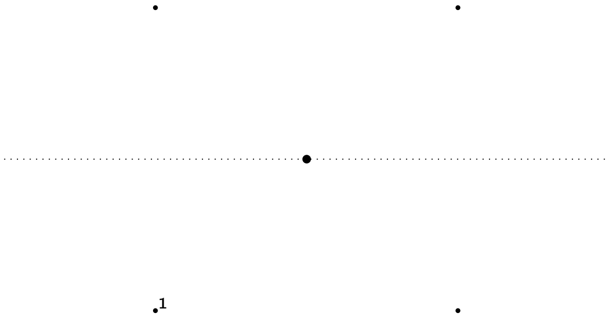
- Nýtum okkur þetta tækifæri og skoðum hvernig við getum útfært rúmfræði í heiltölum.
- Helsti kosturinn við þessa aðferð er að við getum leyst dæmið án fleytitöluskekkju.
- Við geymum þá punkt sem tvennd af heiltölum.
- Við þurfum að geta raðað punktunum eftir stefnuhorni.
- Við getum gert þetta án þess að reikna stefnuhornið.
- Köllum vendipunktinn P og punktanna sem við viljum bera saman X og Y.
- Berum fyrst saman X og Y eftir því hvort þeir séu fyrir ofan eða neðan vendipunktinn.

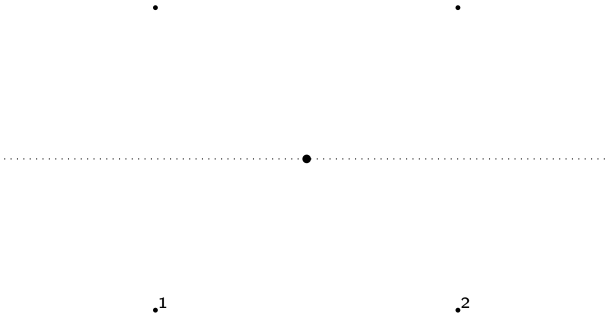
- Nýtum okkur þetta tækifæri og skoðum hvernig við getum útfært rúmfræði í heiltölum.
- Helsti kosturinn við þessa aðferð er að við getum leyst dæmið án fleytitöluskekkju.
- Við geymum þá punkt sem tvennd af heiltölum.
- Við þurfum að geta raðað punktunum eftir stefnuhorni.
- Við getum gert þetta án þess að reikna stefnuhornið.
- Köllum vendipunktinn P og punktanna sem við viljum bera saman X og Y.
- Berum fyrst saman X og Y eftir því hvort þeir séu fyrir ofan eða neðan vendipunktinn.
- Punktar fyrir ofan fá minni forgang.

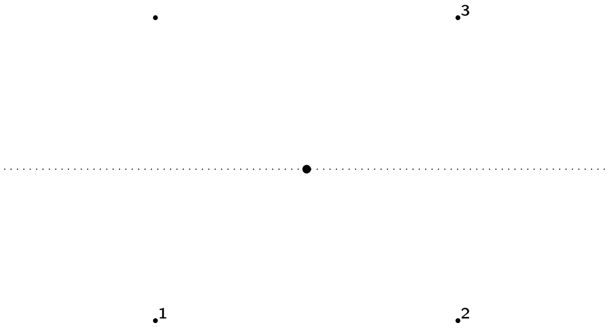
- Til að bera saman punktana X og Y ef þeir eru báðir fyrir ofan eða neðan vendipunktinn þá athugum við í hvaða átt við beygjum þegar við löbbum frá X til Y í gegnum punktinn P.

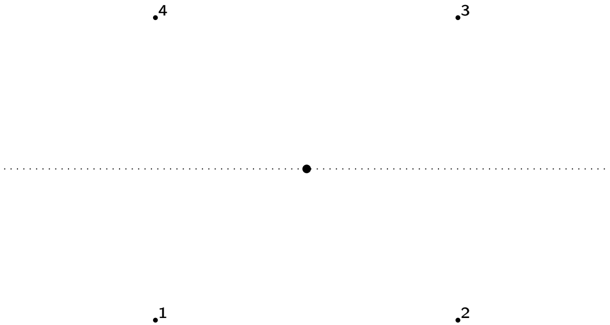
- Til að bera saman punktana X og Y ef þeir eru báðir fyrir ofan eða neðan vendipunktinn þá athugum við í hvaða átt við beygjum þegar við löbbum frá X til Y í gegnum punktinn P.
- Punkturinn X hefur þá minni forgang ef beygjan er til hægri.




1


1
2


3
1
2


4
3
1
2

```c
typedef struct { ll x, y; } pt;
pt topt(ll x, ll y) { pt r = {x, y}; return r; }
ll ccw(pt a, pt b, pt c)
{
    ll r = a.x*(b.y - c.y) + b.x*(c.y - a.y) + c.x*(a.y - b.y);
    if (r == 0) return r;
    return r < 0 ? 1 : -1;
}

ll above(pt a)
{
    if (a.x == 0 && a.y == 0) return 2;
    return a.y < 0 ? 0 : 1;
}

int cmp(const void *p1, const void *p2)
{
    pt a = *(pt*)p1, b = *(pt*)p2;
    ll c = above(a), d = above(b);
    return c != d ? c - d : ccw(a, topt(0, 0), b);
}
```

```c
int main()
{
    ll i, j, k, n, r;
    scanf("%lld", &n);
    while (n)
    {
        pt a[n], b[n];
        for (i = 0; i < n; i++) scanf("%lld%lld", &a[i].x, &a[i].y);
        if (n == 1) { printf("1\n"); scanf("%lld", &n); continue; }
        for (r = 2, i = 0; i < n; i++)
        {
            for (j = 0; j < n; j++)
                b[j].x = a[j].x - a[i].x, b[j].y = a[j].y - a[i].y;
            qsort(b, n, sizeof *b, cmp);
            for (k = 2, j = 1; j < n - 1; j++)
                (ccw(b[j], b[j - 1], b[n - 1]) != 0)
                    ? (k = 2) : (r = max(r, ++k));
        }
        printf("%d\n", r);
        scanf("%lld", &n);
    }
    return 0;
}
```

- Við framkvæmum n sópa, einn fyrir hvern punkt.

- Við framkvæmum n sópa, einn fyrir hvern punkt.
- Í hverjum sóp röðum við punktasafninu, og löbbu svo í gegnum það einu sinni.

- Við framkvæmum n sópa, einn fyrir hvern punkt.
- Í hverjum sóp röðum við punktasafninu, og löbbu svo í gegnum það einu sinni.
- Svo tímaflækjan á hverjum sóp er $\mathcal{O}(\qquad)$.

- Við framkvæmum n sópa, einn fyrir hvern punkt.
- Í hverjum sóp röðum við punktasafninu, og löbbu svo í gegnum það einu sinni.
- Svo tímaflækjan á hverjum sóp er $\mathcal{O}(n \log n)$.

- Við framkvæmum n sópa, einn fyrir hvern punkt.
- Í hverjum sóp röðum við punktasafninu, og löbbu svo í gegnum það einu sinni.
- Svo tímaflækjan á hverjum sóp er $\mathcal{O}(n \log n)$.
- Svo tímaflækjan í heildina er $\mathcal{O}(\qquad)$.

- Við framkvæmum n sópa, einn fyrir hvern punkt.
- Í hverjum sóp röðum við punktasafninu, og löbbu svo í gegnum það einu sinni.
- Svo tímaflækjan á hverjum sóp er $\mathcal{O}(n \log n)$.
- Svo tímaflækjan í heildina er $\mathcal{O}(n^2 \log n)$.

- Skoðum aðra aðferð.

- Skoðum aðra aðferð.
- Látum L_{jk} tákna línuna gegnum j-ta og k-ta punktinn, $j < k$.

- Skoðum aðra aðferð.
- Látum L_{jk} tákna línuna gegnum j-ta og k-ta punktinn, $j < k$.
- Látum svo S_{jk}, $j < k$, tákna fjölda tvennda (j_0, k_0) þannig að $j_0 < k_0$ og $L_{j_0 k_0} = L_{jk}$.

- Skoðum aðra aðferð.
- Látum L_{jk} tákna línuna gegnum j-ta og k-ta punktinn, $j < k$.
- Látum svo S_{jk}, $j < k$, tákna fjölda tvennda (j_0, k_0) þannig að $j_0 < k_0$ og $L_{j_0 k_0} = L_{jk}$.
- Með öðrum orðum táknar S_{jk} hversu mörg eintök eru af línunni L_{jk}.

- Skoðum aðra aðferð.
- Látum L_{jk} tákna línuna gegnum j-ta og k-ta punktinn, $j < k$.
- Látum svo S_{jk}, $j < k$, tákna fjölda tvennda (j_0, k_0) þannig að $j_0 < k_0$ og $L_{j_0 k_0} = L_{jk}$.
- Með öðrum orðum táknar S_{jk} hversu mörg eintök eru af línunni L_{jk}.
- Línan sem flestir punktar liggja er einnig línan sem kemur oftast fyrir.

- Skoðum aðra aðferð.
- Látum L_{jk} tákna línuna gegnum j-ta og k-ta punktinn, $j < k$.
- Látum svo S_{jk}, $j < k$, tákna fjölda tvennda (j_0, k_0) þannig að $j_0 < k_0$ og $L_{j_0 k_0} = L_{jk}$.
- Með öðrum orðum táknar S_{jk} hversu mörg eintök eru af línunni L_{jk}.
- Línan sem flestir punktar liggja er einnig línan sem kemur oftast fyrir.
- Við getum fundið þá línu með því að raða línunum og telja.

- Tökum eftir að ef k punktar liggja á línunni L_{jk} þá er $S_{jk} = k(k-1)/2$.

- Tökum eftir að ef k punktar liggja á línunni L_{jk} þá er $S_{jk} = k(k-1)/2$.
- Svo ef við leysum þessa jöfnu fæst að svarið er $(1+\sqrt{1+8M})/2$ þar sem M er stærsta gildið á S_{jk} sem við getum fengið.

```c
typedef struct { double x, s; } lina;
lina tolina(double x, double s) { lina r = {x, s}; return r; }
int cmp(const void *p1, const void *p2)
{
    lina a = *(lina*)p1, b = *(lina*)p2;
    if (fabs(a.s - b.s) < EPS && fabs(a.x - b.x) < EPS) return 0;
    if (fabs(a.s - b.s) < EPS) return a.x < b.x ? -1 : 1;
    return a.s < b.s ? -1 : 1;
}
```

```c
int main()
{
    ll i, j, n;
    scanf("%lld", &n);
    while (n)
    {
        double x[n], y[n];
        ll k = 0, r = 0;
        lina a[n*n];
        for (i = 0; i < n; i++) scanf("%lf%lf", &x[i], &y[i]);
        if (n == 1) { printf("1\n"); scanf("%lld", &n); continue; }
        for (i = 0; i < n; i++) for (j = 0; j < i; j++)
        {
            if (fabs(x[i] - x[j]) < EPS) a[k++] = tolina(x[i], 1e18);
            else
            {
                double s = (y[i] - y[j])/(x[i] - x[j]);
                a[k++] = tolina(y[i] - s*x[i], s);
            }
        }
        qsort(a, k, sizeof *a, cmp);
        i = 0;
        while (i < k)
        {
            j = i;
            while (j < k && fabs(a[i].s - a[j].s) < EPS
                    && fabs(a[i].x - a[j].x) < EPS) j++;
            r = r < j - i ? j - i : r;
            i = j;
        }
        printf("%d\n", (int)(1.0 + 0.5*sqrt(1.0 + 8.0*r)));
        scanf("%lld", &n);
    }
    return 0;
}
```

- Við höfum $\mathcal{O}(n^2)$ línur sem við þurfum að raða.

- Við höfum $\mathcal{O}(n^2)$ línur sem við þurfum að raða.
- Svo tímaflækjan á þessari lausn er $\mathcal{O}(\qquad\qquad)$.

- Við höfum $\mathcal{O}(n^2)$ línur sem við þurfum að raða.
- Svo tímaflækjan á þessari lausn er $\mathcal{O}(n^2 \log n)$.