**1.**
Konvexní obálku množiny *n* bodů v rovině lze sestrojit

- ✓ vždy se složitosti O(*n* log (*n*))
- o vždy se složitosti $\Omega(n^2)$
- o vždy se složitosti $\Theta(n \log (n))$
- o vždy se složitosti $\Omega(n*k^2)$, kde *k* je počet bodů na konvexní obálce

Pro sestrojení konvexní obálky je v (standardním) Grahamově algoritmu nutno nejprve body seřadit podle jedné ze souřadnic. To zabere v nejhorším případě $\Theta(n \cdot \log (n))$ času. Dále se již všechny seřazené body jednou projdou a každý se maximálně jednou do obálky zařadí a maximálně jednou z ní vyjme, což jsou akce konstantní složitosti, čili v nejhorším případě vše proběhne v čase úměrném $const \cdot n + n \cdot \log (n)$ tj. $n \cdot \log (n)$. Při dodatečné informaci o datech je možno případně použít vhodné řazení pracující v čase $\Theta(n)$, čímž se celková asymptotická rychlost ještě sníží. Celkem tedy máme $O(n \cdot \log (n))$, což je první možnost v odpovědích.

**2.**
Konvexní obálka množiny *n* bodů v rovině může být spočtena Jarvisovým algoritmem (gift wrapping)

- o vždy se složitosti $O(n \log (n))$
- o vždy se složitosti $\Omega(n^2)$
- o vždy se složitosti $\Theta(n \log (n))$
- ✓ vždy se složitosti $O(n*k)$, kde *k* je počet bodů na konvexní obálce

Pro nalezení dalšího bodu při konstrukci konvexní obálky pomocí Jarvisova algoritmu je zapotřebí nanejvýš projít všechny dané body a z nich vybrat jediný, který splňuje podmínku „nejmenšího úhlu“ pro otočení „obalující přímky“ (viz. obr. v přednášce apod.). Má-li ovšem obálky jen *k* bodů (např. 3) je nutno celý proces opakovat právě *k* kráte (třikráte). Jedině poslední čtvrtá odpověď vyhovuje této jednoduché úvaze.

**3.**
Zametací technika (plane sweep)

a) je specializovaný postup pro výpočet průsečíků úseček v rovině
b) je programátorské paradigma k řešení úloh v rovině. Výpočet při něm postupuje podle postupového plánu na jedné souřadné ose. Vše nalevo od zametací přímky je vyřešeno, zametací přímka s sebou nese pomocné proměnné, uchovávající potřebné mezivýsledky na výpočet řešení napravo.
c) je obecný postup dělení roviny na oblasti geometricky nejbližší zadaným bodům
d) je obecný přístup k řešení úloh. Zadané hodnoty rekurzivně dělíme na menší skupiny, pro dostatečně malé množiny hodnot vypočítáme mezivýsledek a mezivýsledky slučujeme.

Platí b). Varianta d) popisuje paradigma (postup, strategii nebo metodiku, chcete-li) rozděl a panuj. Varianty a) a c) lze pomocí zametací techniky řešit, ovšem jsou to jen jednotlivé případy.

**4.**
Voronoiův diagram v rovině sestrojený pro *n* bodů

a) má právě *n*-1 konvexních oblastí
b) má vždycky *n* uzavřených konvexních oblastí
c) má právě *n* obvykle konvexních oblastí
d) má právě *n* konvexních oblastí

Libovolný bod X  bod ze zadané množiny $n$ bodů  leží ve vnitřku určité oblasti, uvnitř které všechny ostatní body roviny mají vzdálenost k X menší než vzdálenost ke kterémukoli jinému bodu zadané množiny. Oblastí tedy musí být $n$, možnost a) neplatí. Diagram pokrává celou rovinu, uzavřená oblast není nekonečná, sjednocení $n$ uzavřených oblastí také nemůže být nekonečné, možnost b) neplatí. Oblasti VD jsou konvexní vždy (důkaz?), odpadá i možnost c). Zbývá jen d).

**5.**

Voronoiův diagram v rovině sestrojený pro $n$ bodů

a) má $O(n^2)$ vrcholů a $O(n^2)$ hran
b) má $O(n^2)$ vrcholů a $O(n)$ hran
c) má $O(n)$ vrcholů a $O(n)$ hran
d) má $O(n)$ vrcholů a $O(n^2)$ hran

Kdyby byl počet hran nebo uzlů kvadratický ve vztahu k počtu bodů, nebylo by možné vytvořit celý diagram v čase $O(n \cdot \log_2(n))$ neboť $n^2 \notin O(n \cdot \log_2(n))$. Možnosti a), b) a d) tak odpadají a zbývá jedině c).

**6.**

Konvexní obálka množiny $n$ bodů v rovině

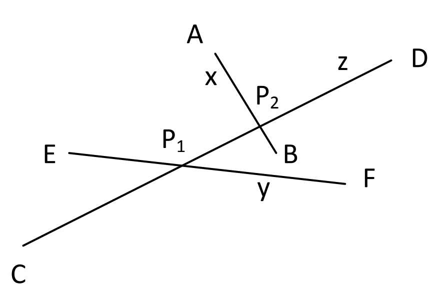
a) je konvexní mnohoúhelník, jehož hrany procházejí všemi zadanými body
b) má vždycky $n$ uzavřených konvexních oblastí
c) je konvexní mnohoúhelník, jehož hrany procházejí všemi zadanými body
d) je nejmenší konvexní mnohoúhelník, který obsahuje všechny zadané body

Konvexní obálka je pojem velmi názorný a intuitivní, stačí pohled na libovolnou definici a jí doprovázející obrázek, abychom nahlédli, že je vyjádřena právě variantou d).

**7.**

Na obrázku je skupina úseček. Při výpočtu jejich vzájemných průsečíků se udržuje postupový plán (x-struktura) a struktura svázaná s přímkou uchovávající mezivýsledky (y-struktura) v pořadí shora. Jaký je stav obou struktur po dokončení 3. kroku podle postupového plánu, tj.po aktualizaci obou struktur?

a) X-str: $P_1$,A, $P_2$,B, F, D
   Y-str: x,z
b) X-str: A, B, F, D
   Y-str: z,y
c) X-str: A, $P_2$,B, F, D
   Y-str: x,z,y
d) X-str: $P_1$, $P_2$,E, F, D
   Y-str: z,y

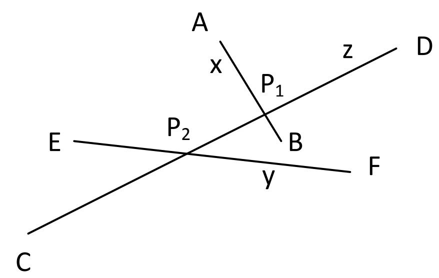



Jednotlivé kroky se činí zleva doprava, přičemž se postupuje po bodech, jenž jsou buďto krajními body úseček, nebo průsečíky, které již algoritmus při svém postupu zleva doprava nalezl. Protože ze začátku nejsou žádné průsečíky známy, obsahuje postupový plán (x-struktura) pouze krajní body úseček – C,E,A,B,F,D. V prvním kroku vstoupí  algoritmus do bodu C a zaregistruje v y-struktuře jemu příslušející úsečku z. Ve druhém kroku vstoupí do bodu E a zaregistruje úsečku y nad(!) úsečkou z. V tu chvíli také určí průsečík úseček z a y – bod $P_1$. Bod $P_1$ se tak stává dalším zastavením postupového plánu zleva doprava a je vložen na jemu příslušné místo: $P_1$, A, B, F, D (nesmíme přitom zapomínat, že body již jednou navštívené se z postupového plánu vylučují). Při zpracování bodu $P_1$, což je třetí krok, se jednak obrátí pořadí úseček v y-struktuře, tj. z se ocitne nad y a dále P je vyloučeno z x-struktury. Žádný nový průsečík nebude objeven. Poté tedy máme: y-struktura v pořadí shora: z,y; x-struktura A,B,F,D. To odpovídá variantě b).

Lze uvažovat i ještě mnohem jednodušeji: Ve třetím kroku algoritmus vstoupí do bodu $P_1$, a tam ještě nemůže objevit bod P2, neboť dosud nedorazil do bodu A, tudíž dosud nemohl zaregistrovat úsečku x i s jejími průsečíky. Bod P2 tedy nemůže bát ještě prvkem postupového plánu (x-struktury). Ve variantách a), c) a d) však uveden je, což je nesprávně, takže zbývá jen varianta b).

**8.**
Na obrázku je skupina úseček. Při výpočtu jejich vzájemných průsečíků se udržuje postupový plán (x-struktura) a struktura svázaná s přímkou uchovávající mezivýsledky (y-struktura) v pořadí shora. Jaký je stav obou struktur po dokončení 4. kroku podle postupového plánu, tj.po aktualizaci obou struktur?

a) X-str: $P_2$,A, $P_1$,B, F, D
   Y-str: z,y,x
b) X-str: $P_1$, $P_2$,B, F, D
   Y-str: z,y
c) X-str:,A, B, F, D
   Y-str: z,x,y
d) X-str: $P_1$,B, F, D
   Y-str: x,z,y




Řešení této úlohy navazuje na řešení předchozí úlohy. Je tam stejný obrázek, jen je prohozeno označení $P_2$ a $P_1$ a zkoumá se stav po třetím kroku. Podle toho tedy stav obou struktur v této úloze po třetím kroku je x-str: A, B, F, D, y-str: z,y. Ve čtvrtém kroku je zaregistrována úsečka x a přidána do y-struktury nahoru a také je díky ní objeven průsečík P1 a přidán do x-struktury. Poté je A z x-struktury vyřazeno. V obou strukturách se tak ocitne právě to, co je popsáno ve variantě d).
Lze také uvažovat jednodušeji. Bod A je čtvrtý zleva, takže po čtvrtém kroku, již nemůže být v x-struktuře, tím odpadají varianty a) a c). Bod $P_2$ je dokonce jen třetí zleva, takže z podobného důvodu odpadá varianta b).

**9.**
Zametací technika (plane sweep)

a) je specializovaný postup pro výpočet průsečíků úseček v rovině
b) je programátorské paradigma k řešení úloh v rovině. Výpočet při něm postupuje podle postupového plánu na jedné souřadné ose. Vše nalevo od zametací přímky je vyřešeno,

zametací přímka s sebou nese pomocné proměnné, uchovávající potřebné mezivýsledky na výpočet řešení napravo.

c) je programátorské paradigma k řešení úloh v rovině. Výpočet při něm postupuje podle postupového plánu na jedné souřadné ose. Vše nalevo od zametací přímky je vyřešeno, zametací přímka s sebou nese seznam všech výsledků, které byly již vypočítány a dále pomocné proměnné, uchovávající potřebné mezivýsledky na výpočet řešení napravo
d) je programátorské paradigma k řešení úloh v rovině. Výpočet při něm postupuje podle předem sestaveného neměnného postupového plánu na jedné souřadné ose. Vše nalevo od zametací přímky je vyřešeno, zametací přímka s sebou nese pomocné proměnné, uchovávající potřebné mezivýsledky na výpočet řešení napravo.

Nuže, zametací technika, jak by již název měl napovídat, není specializovaný postup řešící jednu konkrétní úlohu. Kromě nepravděpodobné varianty a) tu tedy máme tři dlouhatánské kandidáty. Když pomineme jejich společné části, zbyde přibližně:
b) … nese pomocné proměnné, uchovávající potřebné mezivýsledky …
c) … nese seznam všech výsledků…
d) … podle předem sestaveného neměnného postupového plánu…

Varianta d) nemůže být nejspíše pravdivá, neboť kdyby byl postupový plán znám do detailu již od začátku, nebylo by možná již co počítat. Ve variantě c) se navrhuje, že postupující přímka zastupuje skladiště výsledků, což také nepůsobí dobrým dojmem. Zbývá tak jen nejstřízlivěji se tvářící varianta b).