Metódy v bioinformatike

CB #2 Úvod do dynamického programovania

Jana Černíková

FMFI UK

02/10/2025

Problém platenia minimálnym počtom mincí

Vstup: hodnoty k mincí $m_1, m_2, \ldots, m_k$ a cieľová suma X (všetko kladné celé čísla).

Výstup: najmenší počet mincí, ktoré potrebujeme na zaplatenie X.

Problém platenia minimálnym počtom mincí

Vstup: hodnoty k mincí $m_1, m_2, \ldots, m_k$ a cieľová suma X
(všetko kladné celé čísla).
Výstup: najmenší počet mincí, ktoré potrebujeme na zaplatenie X.

Príklad: $k = 3$, $m_1 = 1$, $m_2 = 2$, $m_3 = 5$, $X = 13$.

Odbočka: ešte matematickejšia formulácia bez slov minca, suma, ...

Vstup: kladné celé čísla $m_1, m_2, \ldots, m_k$ a X.
Výstup: celé číslo n a n čísel $x_1, \ldots, x_n$, pre ktoré platia nasledujúce podmienky:

- $x_i \in \{m_1, m_2, \ldots, m_k\}$ pre každé $i = 1, 2, \ldots, n$.
- $\sum_{i=1}^{n} x_i = X$.
- n je najmenšie možné.

Problém platenia minimálnym počtom mincí

Vstup: hodnoty k mincí $m_1, m_2, \ldots, m_k$ a cieľová suma X (všetko kladné celé čísla).
Výstup: najmenší počet mincí, ktoré potrebujeme na zaplatenie X.

Príklad: $k = 3,\ m_1 = 1,\ m_2 = 2,\ m_3 = 5,\ X = 13.$
Príklad: $k = 3,\ m_1 = 1,\ m_2 = 3,\ m_3 = 4,\ X = 6.$

Jednoduchý spôsob riešenia

- opakuj kým $X > 0$:
  - použi najväčšiu mincu, ktorá je najviac X
  - odčítaj hodnotu mince od X

Jednoduchý spôsob riešenia

- opakuj kým $X > 0$:
  - použi najväčšiu mincu, ktorá je najviac X
  - odčítaj hodnotu mince od X

príklad pre $k = 3$, $m_1 = 1$, $m_2 = 2$, $m_3 = 5$, $X = 13$:

Jednoduchý spôsob riešenia

- opakuj kým $X > 0$:
  - použi najväčšiu mincu, ktorá je najviac X
  - odčítaj hodnotu mince od X

príklad pre $k = 3$, $m_1 = 1$, $m_2 = 2$, $m_3 = 5$, $X = 13$:

- použijeme 5, X = 8

Jednoduchý spôsob riešenia

- opakuj kým $X > 0$:
  - použi najväčšiu mincu, ktorá je najviac X
  - odčítaj hodnotu mince od X

príklad pre $k = 3$, $m_1 = 1$, $m_2 = 2$, $m_3 = 5$, $X = 13$:

- použijeme 5, X = 8
- použijeme 5, X = 3

Jednoduchý spôsob riešenia

- opakuj kým $X > 0$:
  - použi najväčšiu mincu, ktorá je najviac X
  - odčítaj hodnotu mince od X

príklad pre $k = 3$, $m_1 = 1$, $m_2 = 2$, $m_3 = 5$, $X = 13$:

- použijeme 5, X = 8
- použijeme 5, X = 3
- použijeme 2, X = 1

Jednoduchý spôsob riešenia

- opakuj kým $X > 0$:
  - použi najväčšiu mincu, ktorá je najviac X
  - odčítaj hodnotu mince od X

príklad pre $k = 3$, $m_1 = 1$, $m_2 = 2$, $m_3 = 5$, $X = 13$:

- použijeme 5, X = 8
- použijeme 5, X = 3
- použijeme 2, X = 1
- použijeme 1, X = 0

Jednoduchý spôsob riešenia

- opakuj kým $X > 0$:
  - použi najväčšiu mincu, ktorá je najviac X
  - odčítaj hodnotu mince od X

príklad pre $k = 3$, $m_1 = 1$, $m_2 = 2$, $m_3 = 5$, $X = 13$:

- použijeme 5, X = 8
- použijeme 5, X = 3
- použijeme 2, X = 1
- použijeme 1, X = 0

Problém s týmto riešením?

Jednoduchý spôsob riešenia

- opakuj kým $X > 0$:
  - použi najväčšiu mincu, ktorá je najviac X
  - odčítaj hodnotu mince od X

príklad pre $k = 3$, $m_1 = 1$, $m_2 = 2$, $m_3 = 5$, $X = 13$:

- použijeme 5, X = 8
- použijeme 5, X = 3
- použijeme 2, X = 1
- použijeme 1, X = 0

Problém s týmto riešením? nefunguje vždy

Jednoduchý spôsob riešenia

- opakuj kým $X > 0$:
  - použi najväčšiu mincu, ktorá je najviac X
  - odčítaj hodnotu mince od X

Problém s týmto riešením: nefunguje vždy

Jednoduchý spôsob riešenia

- opakuj kým $X > 0$:
  - použi najväčšiu mincu, ktorá je najviac X
  - odčítaj hodnotu mince od X

Problém s týmto riešením: nefunguje vždy

príklad:

- mince hodnôt 1,3,4
- X = 6

Jednoduchý spôsob riešenia

- opakuj kým $X > 0$:
  - použi najväčšiu mincu, ktorá je najviac X
  - odčítaj hodnotu mince od X

Problém s týmto riešením: nefunguje vždy

príklad:

- mince hodnôt 1,3,4
- X = 6
- algoritmus:

Jednoduchý spôsob riešenia

- opakuj kým $X > 0$:
  - použi najväčšiu mincu, ktorá je najviac X
  - odčítaj hodnotu mince od X

Problém s týmto riešením: nefunguje vždy

príklad:

- mince hodnôt 1,3,4
- X = 6
- algoritmus: $4 + 1 + 1$

Jednoduchý spôsob riešenia

- opakuj kým $X > 0$:
  - použi najväčšiu mincu, ktorá je najviac X
  - odčítaj hodnotu mince od X

Problém s týmto riešením: nefunguje vždy

príklad:

- mince hodnôt 1,3,4
- X = 6
- algoritmus: $4 + 1 + 1$
- optimum:

Jednoduchý spôsob riešenia

- opakuj kým $X > 0$:
  - použi najväčšiu mincu, ktorá je najviac X
  - odčítaj hodnotu mince od X

Problém s týmto riešením: nefunguje vždy

príklad:

- mince hodnôt 1,3,4
- X = 6
- algoritmus: $4 + 1 + 1$
- optimum: $3 + 3$

Riešenie dynamickým programovaním

- zrátame najlepší počet mincí nielen pre X, ale pre všetky možné cieľové sumy $1, 2, 3, ..., X-1, X$
- vyrobíme si tabuľku A, do ktorej si pre všetky sumy $i = 1, 2, 3, ..., X-1, X$ uložíme najmenší počet mincí, ktorými ich vieme zaplatiť
  - A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

Riešenie dynamickým programovaním

- zrátame najlepší počet mincí nielen pre X, ale pre všetky možné cieľové sumy $1, 2, 3, ..., X-1, X$
- vyrobíme si tabuľku A, do ktorej si pre všetky sumy $i = 1, 2, 3, ..., X-1, X$ uložíme najmenší počet mincí, ktorými ich vieme zaplatiť
  - A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9
$A[i]$										

Riešenie dynamickým programovaním

- zrátame najlepší počet mincí nielen pre X, ale pre všetky možné cieľové sumy $1, 2, 3, ..., X-1, X$
- vyrobíme si tabuľku A, do ktorej si pre všetky sumy $i = 1, 2, 3, ..., X-1, X$ uložíme najmenší počet mincí, ktorými ich vieme zaplatiť
  - A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9
$A[i]$	0									

Riešenie dynamickým programovaním

- zrátame najlepší počet mincí nielen pre X, ale pre všetky možné cieľové sumy $1, 2, 3, ..., X-1, X$
- vyrobíme si tabuľku A, do ktorej si pre všetky sumy $i = 1, 2, 3, ..., X-1, X$ uložíme najmenší počet mincí, ktorými ich vieme zaplatiť
  - A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9
$A[i]$	0	1								

Riešenie dynamickým programovaním

- zrátame najlepší počet mincí nielen pre X, ale pre všetky možné cieľové sumy $1, 2, 3, ..., X-1, X$
- vyrobíme si tabuľku A, do ktorej si pre všetky sumy $i = 1, 2, 3, ..., X-1, X$ uložíme najmenší počet mincí, ktorými ich vieme zaplatiť
  - A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9
$A[i]$	0	1	2							

Riešenie dynamickým programovaním

- zrátame najlepší počet mincí nielen pre X, ale pre všetky možné cieľové sumy $1, 2, 3, ..., X-1, X$
- vyrobíme si tabuľku A, do ktorej si pre všetky sumy $i = 1, 2, 3, ..., X-1, X$ uložíme najmenší počet mincí, ktorými ich vieme zaplatiť
  - A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9
$A[i]$	0	1	2	1						

Riešenie dynamickým programovaním

- zrátame najlepší počet mincí nielen pre X, ale pre všetky možné cieľové sumy $1, 2, 3, ..., X-1, X$
- vyrobíme si tabuľku A, do ktorej si pre všetky sumy $i = 1, 2, 3, ..., X-1, X$ uložíme najmenší počet mincí, ktorými ich vieme zaplatiť
  - A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9
$A[i]$	0	1	2	1	1					

Riešenie dynamickým programovaním

- zrátame najlepší počet mincí nielen pre X, ale pre všetky možné cieľové sumy $1, 2, 3, ..., X-1, X$
- vyrobíme si tabuľku A, do ktorej si pre všetky sumy $i = 1, 2, 3, ..., X-1, X$ uložíme najmenší počet mincí, ktorými ich vieme zaplatiť
  - A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9
$A[i]$	0	1	2	1	1	2				

Riešenie dynamickým programovaním

- zrátame najlepší počet mincí nielen pre X, ale pre všetky možné cieľové sumy $1, 2, 3, ..., X-1, X$
- vyrobíme si tabuľku A, do ktorej si pre všetky sumy $i = 1, 2, 3, ..., X-1, X$ uložíme najmenší počet mincí, ktorými ich vieme zaplatiť
  - A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9
$A[i]$	0	1	2	1	1	2	2			

Riešenie dynamickým programovaním

- zrátame najlepší počet mincí nielen pre X, ale pre všetky možné cieľové sumy $1, 2, 3, ..., X-1, X$
- vyrobíme si tabuľku A, do ktorej si pre všetky sumy $i = 1, 2, 3, ..., X-1, X$ uložíme najmenší počet mincí, ktorými ich vieme zaplatiť
  - A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9
$A[i]$	0	1	2	1	1	2	2	2		

Riešenie dynamickým programovaním

- zrátame najlepší počet mincí nielen pre X, ale pre všetky možné cieľové sumy $1, 2, 3, ..., X-1, X$
- vyrobíme si tabuľku A, do ktorej si pre všetky sumy $i = 1, 2, 3, ..., X-1, X$ uložíme najmenší počet mincí, ktorými ich vieme zaplatiť
  - A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9
$A[i]$	0	1	2	1	1	2	2	2	2	

Riešenie dynamickým programovaním

- zrátame najlepší počet mincí nielen pre X, ale pre všetky možné cieľové sumy $1, 2, 3, ..., X-1, X$
- vyrobíme si tabuľku A, do ktorej si pre všetky sumy $i = 1, 2, 3, ..., X-1, X$ uložíme najmenší počet mincí, ktorými ich vieme zaplatiť
  - A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9
$A[i]$	0	1	2	1	1	2	2	2	2	3

Riešenie dynamickým programovaním

A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	?

X = 10, mince: 1, 3, 4

prvá minca	1	3	4

Riešenie dynamickým programovaním

A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	?

X = 10, mince: 1, 3, 4

prvá minca	1	3	4
X - prvá			

Riešenie dynamickým programovaním

A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	?

X = 10, mince: 1, 3, 4

prvá minca	1	3	4
X - prvá	10 - 1 = 9		

Riešenie dynamickým programovaním

A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	?

X = 10, mince: 1, 3, 4

prvá minca	1	3	4
X - prvá	10 - 1 = 9	10 - 3 = 7	

Riešenie dynamickým programovaním

A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	?

X = 10, mince: 1, 3, 4

prvá minca	1	3	4
X - prvá	10 - 1 = 9	10 - 3 = 7	10 - 4 = 6

Riešenie dynamickým programovaním

A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	?

X = 10, mince: 1, 3, 4

prvá minca	1	3	4
X - prvá	10 - 1 = 9	10 - 3 = 7	10 - 4 = 6
# mincí ešte potrebujeme			

Použijeme tabuľku

Riešenie dynamickým programovaním

A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	?

X = 10, mince: 1, 3, 4

prvá minca	1	3	4
X - prvá	10 - 1 = 9	10 - 3 = 7	10 - 4 = 6
# mincí ešte potrebujeme	3		

Použijeme tabuľku

Riešenie dynamickým programovaním

A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	?

X = 10, mince: 1, 3, 4

prvá minca	1	3	4
X - prvá	10 - 1 = 9	10 - 3 = 7	10 - 4 = 6
# mincí ešte potrebujeme	3	2	

Použijeme tabuľku

Riešenie dynamickým programovaním

A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	?

X = 10, mince: 1, 3, 4

prvá minca	1	3	4
X - prvá	10 - 1 = 9	10 - 3 = 7	10 - 4 = 6
# mincí ešte potrebujeme	3	2	2

Použijeme tabuľku

Riešenie dynamickým programovaním

A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	?

X = 10, mince: 1, 3, 4

prvá minca	1	3	4
X - prvá	10 - 1 = 9	10 - 3 = 7	10 - 4 = 6
# mincí ešte potrebujeme	3	2	2
dokopy mincí			

Riešenie dynamickým programovaním

A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	?

X = 10, mince: 1, 3, 4

prvá minca	1	3	4
X - prvá	10 - 1 = 9	10 - 3 = 7	10 - 4 = 6
# mincí ešte potrebujeme	3	2	2
dokopy mincí	4		

Riešenie dynamickým programovaním

A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	?

X = 10, mince: 1, 3, 4

prvá minca	1	3	4
X - prvá	10 - 1 = 9	10 - 3 = 7	10 - 4 = 6
# mincí ešte potrebujeme	3	2	2
dokopy mincí	4	3	

Riešenie dynamickým programovaním

A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	?

X = 10, mince: 1, 3, 4

prvá minca	1	3	4
X - prvá	10 - 1 = 9	10 - 3 = 7	10 - 4 = 6
# mincí ešte potrebujeme	3	2	2
dokopy mincí	4	3	3

Riešenie dynamickým programovaním

A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	3

X = 10, mince: 1, 3, 4

prvá minca	1	3	4
X - prvá	10 - 1 = 9	10 - 3 = 7	10 - 4 = 6
# mincí ešte potrebujeme	3	2	2
dokopy mincí	4	3	3

Riešenie dynamickým programovaním

A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	3

X = 10, mince: 1, 3, 4

prvá minca	1	3	4
X - prvá	10 - 1 = 9	10 - 3 = 7	10 - 4 = 6
# mincí ešte potrebujeme	3	2	2
dokopy mincí	4	3	3

$$A[10] = min\{A[9] + 1, A[7] + 1, A[6] + 1\}$$

Riešenie dynamickým programovaním

A[i] = najmenší počet mincí, ktoré treba na zaplatenie sumy i

mince 1, 3, 4

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	3

X = 10, mince: 1, 3, 4

prvá minca	1	3	4
X - prvá	10 - 1 = 9	10 - 3 = 7	10 - 4 = 6
# mincí ešte potrebujeme	3	2	2
dokopy mincí	4	3	3

$$A[10] = min\{A[9] + 1, A[7] + 1, A[6] + 1\}$$

$$A[i] = min\{A[i-1] + 1, A[i-3] + 1, A[i-4] + 1\}$$

Algoritmus pre všeobecnú sústavu k mincí $m_1, m_2, \ldots, m_k$

Podproblém A[i]

$$A[i] = 1 + \min\{A[i - m_1], A[i - m_2], \ldots, A[i - m_k]\}$$

```
m = [1,3,4]
X = 11
k = len(m)
nekonecno = math.inf
A = [0]
for i in range(1, X + 1):
    min = nekonecno
    for j in range(k):
        if i >= m[j] and A[i - m[j]] < min:
            min = A[i - m[j]]
    A.append(1 + min)
print(A)
```

Ako nájsť ktoré mince použiť?

Pridáme druhú tabuľku B, kde v B[i] si pamätáme, ktorá bola najlepšia prvá minca, keď sme počítali A[i]
(ak je viac možností, zoberieme ľubovoľnú, napr. najväčšiu)

Ako nájsť ktoré mince použiť?

Pridáme druhú tabuľku B, kde v B[i] si pamätáme, ktorá bola najlepšia prvá minca, keď sme počítali A[i]
(ak je viac možností, zoberieme ľubovoľnú, napr. najväčšiu)

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	3
$B[i]$	-	1	1	3	4	4	3	4	4	4	4

Ako nájsť ktoré mince použiť?

Pridáme druhú tabuľku B, kde v B[i] si pamätáme, ktorá bola najlepšia prvá minca, keď sme počítali A[i]
(ak je viac možností, zoberieme ľubovoľnú, napr. najväčšiu)

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	3
$B[i]$	-	1	1	3	4	4	3	4	4	4	4

Rekonštrukcia riešenia pre sumu 10:

Ako nájsť ktoré mince použiť?

Pridáme druhú tabuľku B, kde v B[i] si pamätáme, ktorá bola najlepšia prvá minca, keď sme počítali A[i]
(ak je viac možností, zoberieme ľubovoľnú, napr. najväčšiu)

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	3
$B[i]$	-	1	1	3	4	4	3	4	4	4	4

Rekonštrukcia riešenia pre sumu 10:

- B[10] = 4, zostane nám zaplatiť 6

Ako nájsť ktoré mince použiť?

Pridáme druhú tabuľku B, kde v B[i] si pamätáme, ktorá bola najlepšia prvá minca, keď sme počítali A[i]
(ak je viac možností, zoberieme ľubovoľnú, napr. najväčšiu)

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	3
$B[i]$	-	1	1	3	4	4	3	4	4	4	4

Rekonštrukcia riešenia pre sumu 10:

- B[10] = 4, zostane nám zaplatiť 6
- B[6] = 3, zostane nám zaplatiť 3

Ako nájsť ktoré mince použiť?

Pridáme druhú tabuľku B, kde v B[i] si pamätáme, ktorá bola najlepšia prvá minca, keď sme počítali A[i]
(ak je viac možností, zoberieme ľubovoľnú, napr. najväčšiu)

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	3
$B[i]$	-	1	1	3	4	4	3	4	4	4	4

Rekonštrukcia riešenia pre sumu 10:

- B[10] = 4, zostane nám zaplatiť 6
- B[6] = 3, zostane nám zaplatiť 3
- B[3] = 3, zostáva 0

Ako nájsť ktoré mince použiť?

Pridáme druhú tabuľku B, kde v B[i] si pamätáme, ktorá bola najlepšia prvá minca, keď sme počítali A[i]
(ak je viac možností, zoberieme ľubovoľnú, napr. najväčšiu)

i	0	1	2	3	4	5	6	7	8	9	10
$A[i]$	0	1	2	1	1	2	2	2	2	3	3
$B[i]$	-	1	1	3	4	4	3	4	4	4	4

Rekonštrukcia riešenia pre sumu 10:

- B[10] = 4, zostane nám zaplatiť 6
- B[6] = 3, zostane nám zaplatiť 3
- B[3] = 3, zostáva 0
- riešenie: 4 + 3 + 3

Program aj s výpisom mincí

```
m = [1,3,4]
X = 11
k = len(m)
nekonecno = 1000000
A = [0]
B = [-1]
for i in range(1, X + 1):
    min = nekonecno
    min_minca = -1
    for j in range(k):
        if i >= m[j] and A[i - m[j]] < min:
            min = A[i - m[j]]
            min_minca = m[j]
    A.append(1 + min)
    B.append(min_minca)

while X > 0:
    print(B[X])
    X = X - B[X]
```

Program aj s výpisom mincí - okrajové prípady

```
pre m = [2,4,5], X = 11, k = len(m)
inicilizacie: nekonecno = 1000000, A = [0], B = [-1]
for i in range(1, X + 1):
    min = nekonecno
    min_minca = -1  # ak sa suma i neda zaplatit (sumy 0,1,3),
                    # v cykle sa nenastavi ziadna hodnota
    for j in range(k):
        if i >= m[j] and A[i - m[j]] < min:
            min = A[i - m[j]]
            min_minca = m[j]
    A.append(1 + min)
    B.append(min_minca) #toto by bez tej -1 vyhodilo chybu (not defined)

while X > 0:
    print(B[X]) #vypise 2 4 5;
                #ale napr. pre m = [2,4,6], X = 11 tu program spadne (index -1),
    X = X - B[X]

print(A)
#[0, 1000001, 1, 1000001, 1, 1, 2, 2, 2, 2, 2, 3]
print(B)
#[-1, -1, 2, -1, 4, 5, 2, 2, 4, 4, 5, 2]

atd. (nemusite riesit, kod je len na ukazku :) )
```

Program aj s výpisom mincí - okrajové prípady

```
pre m = [2,4,6], X = 12, k = len(m)
inicilizacie: nekonecno = 1000000, A = [0], B = [-1]
for i in range(1, X + 1):
    min = nekonecno
    min_minca = -1
    for j in range(k):
        if i >= m[j] and A[i - m[j]] < min:
            min = A[i - m[j]]
            min_minca = m[j]
    A.append(1 + min)
    B.append(min_minca)

while X > 0:
    print(B[X]) # vypise 6 6
    X = X - B[X]

print(A)
#[0, 1000001, 1, 1000001, 1, 1000001, 1, 1000001, 2, 1000001, 2, 1000001, 2]
print(B)
#[-1, -1, 2, -1, 4, -1, 6, -1, 2, -1, 4, -1, 6]

atd. (nemusite riesit, kod je len na ukazku :) )
```

Dynamické programovanie vo všeobecnosti

Dynamické programovanie vo všeobecnosti

- Okrem riešenia celého problému riešime aj menšie problémy (nazývame ich podproblémy).

Dynamické programovanie vo všeobecnosti

- Okrem riešenia celého problému riešime aj menšie problémy (nazývame ich podproblémy).
- Riešenia podproblémov ukladáme do tabuľky a používame pri riešení väčších podproblémov.

Dynamické programovanie vo všeobecnosti

- Okrem riešenia celého problému riešime aj menšie problémy (nazývame ich podproblémy).
- Riešenia podproblémov ukladáme do tabuľky a používame pri riešení väčších podproblémov.
- Technika dynamického programovania sa používa na viacero problémov v bioinformatike.
  - napr. hľadanie zarovnaní sekvencií