Téma: **Prevody nezáporných[1] celých čísel medzi pozičnými číselnými sústavami s rôznymi základmi**

Zopakuj si:
Pozičná číselná sústava:

- dekadická (desiatková), so základom 10
  - používa cifry (znaky) 0, 1, ... , 9
  - $1024_{10} = 1.10^3 + 0.10^2 + 2.10^1 + 4.10^0 = 4 + 20 + 0 + 1000 = 1024$
- binárna (dvojková), so základom 2
  - používa len dva znaky 0 a 1
  - $1000001_2 = 1.2^6 + 0.2^5 + 0.2^4 + 0.\ 2^3 + 0.2^2 + 0.\ 2^1 + 1.2^0 = 64 + 1 = 65_{10}$
  - $1111010_2 = 1.2^6 + 1.2^5 + 1.2^4 + 1.\ 2^3 + 0.2^2 + 1.\ 2^1 + 0.2^0 = 64 + 32 + 16 + 8 + 0 + 2 + 0 = 122_{10}$
- hexadecimálna (šestnástková), so základom 16
  - používa 16 znakov: 0, 1, 2,..., 9, A (10_{10}), B (11_{10}), C (12_{10}), D (13_{10}), E (14_{10}) a F (15_{10})
  - $61_{16} = 6.16^1 + 1.16^0 = 96 + 1 = 97$
  - $7A_{16} = 7.16^1 + 10.16^0 = 112 + 10 = 122$
  - $ABBA_{16} = 10.16^3 + 11.16^2 + 11.16^1 + 10.16^0 = 43\ 962$

Python - údajové typy: typ int - Funkcie na prevod medzi číselnými sústavami: bin(i) - vráti binárnu hodnotu celého čísla i; hex(i) - vráti hexadecimálnu (šestnástkovú) hodnotu celého čísla i; oct(i) - vráti oktánovú (osmičkovú) hodnotu celého čísla i; int(reťazec, základ) - prevedie reťazec reprezentujúci celé číslo v číselnej sústave so základom základ na celé číslo alebo vyvolá výnimku ValueError, 2 ≤ základ ≤ 36.

Úloha 1: Vytvorte funkciu na prevod nezáporného celého čísla z desiatkovej do dvojkovej číselnej sústavy.
V pythone funkcia bin(i).

Algoritmus (Informatika pre stredné školy, učebnica pre 1.ročník, Ivan Kalaš a kolektív, SPN, str.34):
1. Desiatkové číslo vydelíme 2.
2. Zapíšeme zvyšok (0 alebo 1).
3. Výsledok delenia opäť vydelíme 2.
4. Zvyšok zapíšeme pred predchádzajúci zvyšok.
5. Opakujeme 3. a 4. krok tak dlho, kým výsledok delenia nie je 0.

```
def preved10do2(cislo10):
    '''prevedie celé nezáporné číslo z desiatkovej do dvojkovej sústavy'''
    if cislo10 == 0: return 0
    str2 = ""
    while cislo10 > 0:
        str2 = str(cislo10%2) + str2
        cislo10 //= 2
    return int(str2)
```

Použitie:
```
cislo10 = 1024
print("Číslo {} so základom 10 = {} so základom 2".format(cislo10,preved10do2(cislo10)))
print("Kontrola: bin(",cislo10,"): ", bin(cislo10), sep="")
```
vypíše:
```
Číslo 1024 so základom 10 = 10000000000 so základom 2
Kontrola: bin(1024): 0b10000000000
```

Úloha 2: Vytvorte funkciu na prevod nezáporného celého čísla z desiatkovej do číselnej sústavy so základom X**,** (2 ≤ X ≤ 10).
Poznámka: Pre X = 10 musíte dostať rovnaké číslo ako zadané číslo.
Algoritmus pre prevod do základu X (2 ≤ X ≤ 10) je zovšeobecnením algoritmu z úlohy 1.

[1] Prevod záporného celého čísla je analogický (rovnaký), ako nezáporného, len výsledkom je záporné celé číslo.

```
def preved10doX(cislo10, zakladX):
    '''
    prevedie celé nezáporné číslo z desiatkovej sústavy
    do sústavy so základom X (2<= X <= 10)
    '''
    if cislo10 == 0: return 0
    strX = ""
    while cislo10 > 0:
        strX = str(cislo10%zakladX) + strX
        cislo10 //= zakladX
    return int(strX)
```

Použitie:

```
cislo10 = 568
zakladX = 8
print("Číslo {} so základom 10 = {} so základom {}".format(cislo10, preved10doX(cislo10,zakladX),zakladX))
print("Kontrola: oct(",cislo10,") = ", oct(cislo10), sep="")
```

vypíše:

```
Číslo 568 so základom 10 = 1070 so základom 8
Kontrola: oct(568) = 0o1070
```

Úloha 3: Vytvorte funkciu na prevod nezáporného celého čísla z desiatkovej do šestnástkovej číselnej sústavy.

V algoritme musíme, navyše od ostatného, vyriešiť konverziu zvyškov 10, 11,..., 15 na znaky A, B,..., F, čo nie je až taký problém.

```
def preved10do16(cislo10):
    '''prevedie celé nezáporné číslo z desiatkovej do šestnástkovej sústavy'''
    if cislo10 == 0: return 0
    ZNAKY16 = "ABCDEF"
    str16 = ""
    while cislo10 > 0:
        zvysok = cislo10%16
        if zvysok <= 9:
            znak = str(zvysok)
        else:
            ''' "klasické" riešenie:
            if zvysok == 10: znak = 'A'
            elif zvysok == 11: znak = 'B'
            elif zvysok == 12: znak = 'C'
            elif zvysok == 13: znak = 'D'
            elif zvysok == 14: znak = 'E'
            else: znak = 'F'
            '''
            znak = ZNAKY16[zvysok-10]              # Využitie konštanty ZNAKY16
        str16 = znak + str16
        cislo10 //= 16
    return str16
```

Podobné riešenie toho istého problému žiačky Natálie Holkovej z III.C:

```
...
    if zvysok >= 10:
        str16 = str(chr(ord("A") + zvysok - 10)) + str16
    else:
        str16 = str(zvysok) + str16
    cislo10 //= 16
```

Použitie:

```
cislo10 = 31
print("Číslo {} so základom 10 = {} so základom 16".format(cislo10, preved10do16(cislo10)))
print("Kontrola: hex(",cislo10,") = ", hex(cislo10), sep="")
```

vypíše:

```
Číslo 31 so základom 10 = 1F so základom 16
Kontrola: hex(31) = 0x1f
```

Úloha 4: Vytvorte funkciu na prevod nezáporného celého čísla z dvojkovej do desiatkovej číselnej sústavy.
V pythone funkcia int(reťazec, 2).

Informatika pre stredné školy, učebnica pre 1.ročník, Ivan Kalaš a kolektív, SPN, str.34:
„Opačný prevod je veľmi jednoduchý. Kým pri prevode do dvojkovej sústavy sme desiatkové číslo opakovane delili dvoma a zapisovali zvyšky - cifry, teraz budeme medzivýsledok postupne násobiť dvoma a pričítavať cifry dvojkového čísla. Vychádzame pri tom z takejto myšlienky: Ak má dvojkové číslo napr. 5 cifier a zapíšeme ho ako ABCDE (kde každé písmeno predstavuje 0 alebo 1), potom jeho desiatkový prevod má hodnotu ((((0 + A).2 + B).2 + C).2 + D).2 + E."

Poznámka: Označenie cifier A, B, C, D a E nemá žiaden súvis so značením v šestnástkovej sústave.

Algoritmus:
1. Prvým medzivýsledkom nech je 0.
2. Oddeľme prvú cifru dvojkového čísla.
3. Ak je oddelenou cifrou 0, medzivýsledok vynásobme 2.
4. Ak je oddelenou cifrou 1, medzivýsledok vynásobme 2 a pripočítajme k nemu 1.
5. Kroky 2, 3 a 4 opakujme tak dlho, až minieme všetky cifry daného čísla. Medzivýsledok je vtedy už konečným výsledkom.

```
def preved2do10(str2):
    """prevedie neprázdny reťazec dvojkového čísla na číslo so základom 10"""
    cislo10 = 0
    for znak in str2:
        cislo10 = cislo10*2 + int(znak)
    return cislo10
```

Použitie:
```
str2 = "10010"
if str2 == "":
    print("Číslo so základom 2 - ZADANÝ PRÁZDNY REŤAZEC")
else:
      print("Číslo {} so základom 2 = {} so základom 10".format(str2, preved2do10(str2)))
      print("Kontrola: int(\"",str2,"\",2) = ",int(str2, 2), sep="")
```
vypíše:
```
Číslo 10010 so základom 2 = 18 so základom 10
Kontrola: int("10010",2) = 18
```

Problém prevodu z „X do 10" (a teda aj z „2 do 10") trochu inak:
Prevod nezáporného celého čísla z pozičnej číselnej sústavy so základom X ($2 \le X < 10$) do desiatkovej číselnej sústavy znamená vypočítať hodnotu výrazu $a_n.X^n + a_{n-1}.X^{n-1} + a_{n-2}.X^{n-2} +...+a_1.X^1 + a_0$ pričom a_i (i = n, n-1,..., 1, 0) sú konkrétne číslice z prevádzaného čísla a pri desiatkovom čísle a_0 predstavuje jednotky, a_1 desiatky, a_2 stovky atď. Dá sa dokázať (pozri Hornerova schéma), že výpočet výrazu $a_n.X^n + a_{n-1}.X^{n-1} + a_{n-2}.X^{n-2} +...+a_1.X^1 + a_0$ možno zefektívniť výpočtom výrazu $(...((a_n.X + a_{n-1}).X + a_{n-2}).X +...+ a_1).X + a_0$ resp. výpočtom výrazu $(...(((0.X + a_n).X + a_{n-1}).X + a_{n-2}).X +...+a_1).X + a_0$, pričom pridanú nulu použijeme ako počiatočnú hodnotu výpočtu. Výhodou takéhoto výpočtu je ušetrenie násobenia – nemusíme počítať X^n, X^{n-1},..., X^2, lebo je „skryté" vo vnorených zátvorkách.
Schéma nás vedie k opakovaniu násobenia „čohosi" (čiastkového výsledku, medzivýsledku) číslom X a k pripočítaniu ďalšej číslice (a_n, a_{n-1}, a_{n-2} až a_0). To „čosi" je najprv 0, ktorú vynásobíme X a pripočítame prvú číslicu zľava prevádzaného čísla, opäť vynásobíme X a pripočítame druhú číslicu atď. až kým nepripočítame poslednú číslicu prevádzaného čísla. Čiže <u>opakujeme</u>: nová hodnota ← predchádzajúca hodnota . X + ďalšia číslica.

Úloha 5: Vytvorte funkciu na prevod nezáporného celého čísla zo šestnástkovej do desiatkovej číselnej sústavy.
V pythone funkcia int(reťazec, 16).

Poznámka:
Šestnástková číselná sústava používa šestnásť znakov 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A (10_{10}), B (11_{10}), C (12_{10}), D (13_{10}), E (14_{10}) a F (15_{10}). Základ je 16 a „cifry" A, B, C, D, E a F musíme pri výpočte nahradiť číslami 10, 11, 12, 13, 14 a 15. Preto treba <u>opakovať</u> výpočet:

nová hodnota ← predchádzajúca hodnota . 16 + ďalší šestnástkový znak zmenený na desiatkové číslo!

```
def preved16do10(str16):
    """prevedie neprázdny reťazec šestnástkového čísla na číslo so základom 10"""
    str16 = str16.upper()
    ZNAKY16 = "ABCDEF"
    cislo10 = 0
    for znak in str16:
        if znak in "0123456789": cislo10 = cislo10*16 + int(znak)
        else:
            cislo10 = cislo10*16 + (10 + ZNAKY16.index(znak))
    return cislo10
```

Použitie:

```
str16 = "1F"
if str16 == "":
    print("Číslo so základom 16 - ZADANÝ PRÁZDNY REŤAZEC")
else:
    print("Číslo {} so základom 16 = {} so základom 10".format(str16, preved16do10(str16)))
    print("Kontrola: int(\""+str16+"\",16) =",int(str16, 16))
```

vypíše:

```
Číslo 1F so základom 16 = 31 so základom 10
Kontrola zápisom 0x1F: 31
```

Vetva `else: cislo10 = cislo10*16 + (10 + ZNAKY16.index(znak))`
v ostatnej funkcii nahrádza vetvy

```
        elif znak == "A": cislo10 = cislo10*16 + 10
        elif znak == "B": cislo10 = cislo10*16 + 11
        elif znak == "C": cislo10 = cislo10*16 + 12
        elif znak == "D": cislo10 = cislo10*16 + 13
        elif znak == "E": cislo10 = cislo10*16 + 14
        else:znak == "F": cislo10 = cislo10*16 + 15
```

Úloha 6: Všetky predchádzajúce prevody upravte tak, aby prevádzali aj záporné celé čísla.

Úloha 7a: Vytvorte funkciu na prevod nezáporného celého čísla zo sústavy zo základom X ($2 \le X \le 10$) do desiatkovej číselnej sústavy. V pythone funkcia int(reťazec, X), pričom $2 \le X \le 36$.

```
def prevedXdo10(strX, zakladX):
    """prevedie reťazec čísla so základom X (2<=X<10) na desiatkové číslo"""
    cislo10 = 0
    for znak in strX:
        cislo10 = cislo10*zakladX + int(znak)
    return cislo10
```

Použitie:

```
strX = "1070"
zakladX = 8
print("Verzia bez kontroly:")
if strX == "":
    print("Číslo so základom",zakladX,"- ZADANÝ PRÁZDNY REŤAZEC")
else:
    print("Číslo {} so základom {} = {} so základom 10".format(strX, zakladX, prevedXdo10(strX, zakladX)))
    print("Kontrola: int(\"",strX,"\",",zakladX,") = ",int(strX, zaklad), sep="")
```

vypíše:

```
Verzia bez kontroly:
Číslo 1070 so základom 8 = 568 so základom 10
Kontrola: int("1070",8) = 568
```

Úloha 7b*: Úlohu 7a doplňte o funkciu, ktorá bude kontrolovať, či v prevádzanom čísle nebola použitá nedovolená cifra, t.j. funkcia vráti True, ak všetky cifry čísla so základom X sú menšie ako X, inak vráti False. Funkciu označme `kontrolaCifierZakladuXv1` (v1 na konci názvu znamená verzia 1).

```
def kontrolaCifierZakladuXv1(strX, zakladX):
    """
    funkcia vráti False, ak je v reťazci strX (reťazec čísla so základom X; 2<=X<10)
    nedovolený znak, t.j. znak mimo dovolený interval; inak vráti True
    """
    for znak in strX:
        if not("0" <= znak < str(zakladX)):
            return False
    return True
```

Autorkou výrazu `not("0" <= znak < str(zakladX))` resp. jeho analógie `not(ord("0") <= ord(znak) < ord(str(zakladX)))` je Natália Holková z III.C. Cením si zmysluplné čítanie a nie mechanické osvojovanie študijného textu. Ďakujem!

Použitie:

```
strX = "1070"
zakladX = 8
print("Verzia s kontrolou, či možno výpočet uskutočniť:")
if strX == "":
    print("Číslo so základom",zakladX,"- ZADANÝ PRÁZDNY REŤAZEC!")
else:
    if kontrolaCifierZakladuXv1(strX, zakladX):
        print("Číslo {} so základom {} = {} so základom 10".format(strX, zakladX, prevedXdo10(strX, zakladX)))
        print("Kontrola: int(\"",strX,"\",",zakladX,") = ",int(strX, zakladX), sep="")
```

vypíše:

```
Verzia s kontrolou, či možno výpočet uskutočniť:
Číslo 1070 so základom 8 = 568 so základom 10
Kontrola: int("1070",8) = 568
```

Úloha 7c*: Úlohu 6a doplňte o funkciu, ktorá bude kontrolovať, či v prevádzanom čísle nebola použitá nedovolená cifra (t.j. všetky cifry čísla so základom X sú menšie ako X; 2<=X<10), ak áno, vráti prvý nedovolený znak, inak vráti prázdny reťazec.

```
def kontrolaCifierZakladuX(strX, zakladX):
    """
    funkcia vráti prvý nedovolený znak z reťazca strX (reťazec čísla so základom X;
    2<=X<10), t.j. znak väčší alebo rovný ako základ X; inak vráti prázdny reťazec!
    """
    for znak in strX:
        if znak >= str(zakladX):
            return znak
    return ""

def prevedXdo10sKontrolou(strX, zakladX):
    """
    funkcia vráti výsledok prevodu - desiatkové číslo alebo prvý nedovolený znak
    z reťazca strX (reťazec čísla so základom X; 2 <= X < 10)
    """
    vysledokKontroly = kontrolaCifierZakladuX(strX, zakladX)
    if vysledokKontroly == "":
        # prepis pôvodnej funkcie prevedXdo10(strX, zakladX):
        cislo10 = 0
        for znak in strX:
            cislo10 = cislo10*zakladX + int(znak)
        return cislo10
    else:
        return vysledokKontroly
```

Použitie:

```
strX = "33"
zakladX = 4
print("Verzia s kontrolou (vráti nedovolený znak):")
if strX == "":
```

```
    print("Číslo so základom",zakladX,"- ZADANÝ PRÁZDNY REŤAZEC!")
else:
    vysledok = prevedXdo10sKontrolou(strX, zakladX)
    if type(vysledok) == int:
        print("Číslo {} so základom {} = {} so základom 10".format(strX, zakladX, vysledok))
       print("Kontrola: int(\"",strX,"\",",zakladX,") = ",int(strX, zakladX), sep="")
    else:
        print("Číslo {} so základom {} = PRVÝ NEDOVOLENÝ ZNAK {}!".format(strX, zakladX, vysledok))
```

vypíše:

```
Verzia s kontrolou (vráti nedovolený znak):
Číslo 33 so základom 4 = 15 so základom 10
Kontrola: int("33",4) = 15
```

pre:

```
strX = "187"
zaklad = 8
...
```

vypíše:

```
Verzia s kontrolou (vráti nedovolený znak):
Číslo 187 so základom 8 = PRVÝ NEDOVOLENÝ ZNAK 8!
```

Úloha 8*: Vytvorte funkciu na prevod nezáporného celého čísla zo sústavy zo základom X ($2 \leq X \leq 36$) do desiatkovej číselnej sústavy. V pythone funkcia int(reťazec, X), pričom $2 \leq X \leq 36$.

```
def kontrolaCifierZakladuXuni(strX, zakladX):
    """
    funkcia vráti prvý nedovolený znak z reťazca strX (reťazec čísla so základom X;
    2<=X<=36), t.j. znak väčší alebo rovný ako základ X; inak vráti prázdny reťazec!
    """
    ZNAKY36 = "0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ"
    for znak in strX:
        if ZNAKY36.index(znak) >= zakladX:
            return znak
    return ""

def prevedXdo10uni(strX, zakladX):
    """
    funkcia vráti výsledok prevodu - desiatkové číslo alebo prvý nedovolený znak
    z reťazca strX (reťazec čísla so základom X; 2 <= X <= 36)
    """
    strX = strX.upper()
    vysledok = kontrolaCifierZakladuXuni(strX, zakladX)
    if vysledok != "":
        return vysledok
    else:
        ZNAKY26 = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
        cislo10 = 0
        for znak in strX:
            if znak in "0123456789": cislo10 = cislo10*zakladX + int(znak)
            else: cislo10 = cislo10*zakladX + (10 + ZNAKY26.index(znak))
        return cislo10
```

Použitie:

```
strX = "2Z"
zakladX = 36
if strX == "":
    print("Číslo so základom",zakladX,"- ZADANÝ PRÁZDNY REŤAZEC")
else:
    vysledok = prevedXdo10uni(strX, zakladX)
    print(vysledok)
    if type(vysledok) == str:
        print("Číslo {} so základom {} = PRVÝ NEDOVOLENÝ ZNAK {}".format(strX, zakladX, vysledok))
    else:
        print("Číslo {} so základom {} = {} so základom 10".format(strX, zakladX, vysledok))
```

```
        print("Kontrola: int(\""+strX+"\",",zakladX,") = ",int(strX, zakladX), sep="")
```

vypíše:

```
Číslo 2Z so základom 36 = 107 so základom 10
Kontrola: int("2Z",36) = 107
```

Úloha 9*: Vytvorte funkciu na prevod nezáporného celého čísla z dvojkovej do šestnástkovej sústavy podľa algoritmu nižšie. Vytvorte a využite funkciu, ktorá vráti jeden hexadecimálny znak po dovezení maximálne štvorznakového binárneho čísla do funkcie.

Algoritmus na prevod čísla z dvojkovej do šestnástkovej sústavy:

1. binárne číslo rozdelíme sprava na štvorčíslia (ak treba, doplníme zľava nulami)
2. každé štvorčíslie prepočítame na hexadecimálne číslo, presnejšie na jednu cifru hexadecimálneho čísla
3. vypočítané cifry tvoria hľadané hexadecimálne číslo

Napríklad:

1. 1111010_2 = 0111 1010
2. $111_2 = 1.2^2 + 1.2^1 + 1.2^0$ = **7**; $1010 = 1.2^3 + 0.2^2 + 1.2^1 + 0.2^0 = 10$ a v šestnástkovej sústave sa zapisuje znakom **A**
3. 1100001_2 = $\mathbf{7A}_{16}$