1940223_2022-02-12 (number conversion)

March 28, 2022

AIM: Convert numbers from different number systems.

1 Convert decimal to binary

```
[36]: n = int(input(">> "))
```

```
>> 6
```

```
[37]: def dec_to_bin(n):
          t, b = n, ''
          while t != 0:
              b = str(t % 2) + b
              t = t // 2
          return b
```

```
[38]: dec_to_bin(n)
```

```
[38]: '110'
```

2 Convert binary to decimal

```
[1]: n = input(">> ")
```

```
>> 101101010101010110101101011110101010101
```

```
[2]: def bin_to_dec(n):
         d, multiplier = 0, 1
         for top in range(len(n) - 1, -1, -1):
             if n[top] not in '01': return 'Input not binary!'
             d = d + int(n[top])*multiplier
             multiplier = multiplier*2
         return d
```

```
[3]: bin_to_dec(n)
```

```
[3]: 389413256533
```

3 Base x to base y (both ways conversion)

[2]:

```python
# The buit-in function '.isupper' is not used.
# This is  because it also returns true for special alphabets.
def isUpper(c):
    if ord(c) >= ord('A') and ord(c) <= ord('Z'): return True
    return False
def putDigit(digit):
    c = chr(int(digit) + 55)
    # 10 should become 'A',whose ASCII number is 65. Hence, we add 55.
    if isUpper(c): return c
    # '.isupper' checks if the character is an uppercase alphabet
    else: return "<" + str(digit) + ">"
    # For 10, returns A, for 11, returns B, etc.
def convert(n, homeBase, targetBase):
    n = str(n)
    d, multiplier, top = 0, 1, len(n) - 1
    #------------------------
    # Converting to decimal...
    while top > -1:
        x = n[top]
        """
        Checking for the following:
        - Alphabets (can be digits in large number bases)
        - Digits enclosed in <...> (can be digits in very large number bases)
        - Simple decimal integer digits
        """
        if isUpper(x): x = ord(x) - 55
        # 'A', whose ASCII number is 65, should become 10. Hence, we subtract 55.
        elif x.isnumeric(): x = int(x)
        elif x == '>':
            top = top - 1
            while top > -1 and n[top].isnumeric():
                x =  n[top] + x
                top = top - 1
            if n[top] == '<': x = int(x[:-1]) # Since last element is '>'
            else: return 'Number contains invalid digits.'
        else: return 'Number contains invalid digits.'
        #------------------------
        # Checking if the digit fits the initial number base...
        if x >= homeBase: return "Number does not match base."
        #------------------------
        # Adding to the decimal number being obtained...
        d = d + x*multiplier
        multiplier = multiplier*homeBase
        top = top - 1
    # Converting to target base...
    t = str()
    while d != 0:
        digit = str(d % targetBase)
        # If digit exceeds 10 (due to target base exceeding 10)...
        if int(digit) >= 10: digit = putDigit(digit)
        t = digit + t
        d = d // targetBase
    return t
```

[18]:

```python
print(convert('ANAKIN', 25, 10))
```

```
106809848
```

[ ]:

```python

```