NUMBER SYSTEM CONVERTER USING PYTHON

def menu():  
  
 return ()  
  
  
def validate\_bin(check\_number):  
  
 check\_list = [int(item) for item in (sorted(set(list(str(check\_number)))))]  
  
 for number in check\_list:  
 print (f'checking {number} - {type(number)}')  
 if number not in [0,1]:  
 print (f'invalid binary number')  
 return False  
 return True  
  
def validate\_input(input\_number):  
  
 legal\_char = '0123456789abcdef'  
 for number in input\_number:  
 if number not in legal\_char:  
 return False  
 return True  
  
def validator(input\_number,input\_base,output\_base):  
 if validate\_input(input\_number) and input\_base.isdigit() and output\_base.isdigit():  
  
 if int(input\_base) == 2:  
 if not validate\_bin(input\_number):  
 print ('ERROR: Invalid Binary Number. Must contain 0s or 1s')  
 return False  
  
 if input\_number.isdigit() and input\_number.isalpha():  
 if int(input\_base) != 16:  
 print ('ERROR: Hexadecimal numbers requires base FROM to be 16')  
 return False  
  
 if int(input\_base) == 1 or int(output\_base) == 1:  
 print (f'can not convert to or from Base-1')  
 return False  
 return True  
  
  
  
  
def convert\_number\_system(input\_number, input\_base, output\_base):  
  
  
 #list that holds the numbers to output in the end  
 remainder\_list = []  
  
 #start value for sum\_base\_10. All calculations go through base-10.  
 sum\_base\_10 = 0  
  
 #validate\_input  
  
  
 if output\_base == 2:  
 binary\_repr = bin(input\_number)  
 return (binary\_repr[2:])  
  
  
  
 # we want to convert to base-10 before the actual calculation:  
 elif input\_base != 10:  
  
 # reverse the string to start calculating from the least significant number  
 reversed\_input\_number = input\_number[::-1]  
  
 #check if user typed in letter outside HEX range.  
 hex\_helper\_dict = {'a' : 10 , 'b' : 11 , 'c' : 12 , 'd' : 13 , 'e' : 14 , 'f' : 15}  
  
  
 for index, number in enumerate(reversed\_input\_number):  
 for key,value in hex\_helper\_dict.items():  
 if str(number).lower() == key:  
 number = value  
  
 sum\_base\_10 += (int(number)\*(int(input\_base)\*\*index))  
  
 # if the number is already in Base-10, we can start the conversion  
 elif input\_base == 10:  
 sum\_base\_10 = int(input\_number)  
  
  
 # we loop through until we hit 0. When we hit 0, we have our number.  
 while sum\_base\_10 > 0:  
  
 #find number to pass further down the loop  
 divided = sum\_base\_10// int(output\_base)  
  
 #find remainder to keep  
 remainder\_list.append(str(sum\_base\_10 % int(output\_base)))  
  
 # the new value to send to the next iteration  
 sum\_base\_10 = divided  
  
  
 #fix the list and send a number:  
 return\_number = ''  
  
 # if the user asked for a Hexadesimal output, we need to convert  
 # any number from 10 and up.  
 if output\_base == 16:  
 hex\_dict = {10 : 'a' , 11 : 'b' , 12 : 'c' , 13 : 'd' , 14 : 'e' , 15 : 'f'}  
  
 #loop through remainder\_list and convert 10+ to letters.  
 for index, each in enumerate(remainder\_list):  
 for key, value in hex\_dict.items():  
 if each == str(key):  
 remainder\_list[index] = value  
  
 #return the number:  
 else:  
 for each in remainder\_list[::-1]:  
 return\_number += each  
  
 return (return\_number)  
 #else:  
 #return ('invalid input... Please Try Again')  
  
def execute\_converter():  
  
 user\_number = ''  
 user\_input\_base = ''  
 user\_output\_base = ''  
  
 proceed = 'y'  
  
 while proceed.lower() == 'y':  
 valid\_input = False  
 while valid\_input == False:  
 user\_number = input('\nPlease type the number you wish to convert: ')  
 user\_input\_base = input('Please type the base you wish to convert FROM (e.g. 10): ')  
 user\_output\_base = input('Please type the base you wish to convert TO (e.g. 2): ')  
  
 valid\_input = validator(user\_number,user\_input\_base,user\_output\_base)  
  
 print (f'\nTrying to convert The Number {user\_number} from a Base-{user\_input\_base} to a Base-{user\_output\_base}: ')  
 print (f'>> RESULT: {convert\_number\_system(user\_number, user\_input\_base, user\_output\_base)} <<')  
  
 print (f'\nDo you wish to convert another number? Type y/n: ')  
 proceed = input('')  
  
  
 print (f'quitting converter...')  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 #print the menu:  
 print(menu())  
 #execute the actual converter:  
 execute\_converter()

OUTPUT:

Welcome to the Number Base converter!

The system will ask you for:

- The Number to convert

- What base you want to convert FROM

- What base you want to convert TO

\*\* REMEMBER: All numbers must be integers!(unless HEX) \*\*

----------------------------------------------------------

Please type the number you wish to convert: 29

Please type the base you wish to convert FROM (e.g. 10): 10

Please type the base you wish to convert TO (e.g. 2): 2

Trying to convert The Number 29 from a Base-10 to a Base-2:

>> RESULT: 11101 <<

Do you wish to convert another number? Type y/n:

y

Please type the number you wish to convert: 30

Please type the base you wish to convert FROM (e.g., 10): 10

Please type the base you wish to convert TO (e.g. 2): 8

Trying to convert The Number 30 from a Base-10 to a Base-8:

>> RESULT: 36 <<

Do you wish to convert another number? Type y/n:

n

quitting converter...

Process finished with exit code 0

DESCRIPTION:

The Base converter App In Python is a simple project developed using Python. The project contains a base conversion system and calculator. The user can enter any number and convert them to any number systems (like decimal to binary, binary to octal, etc.). Moreover, the user can also use a calculator from this application.

This is a simple GUI-base project which is very easy to understand and use. Also, this project makes a convenient way for the user to gain an idea of how to perform number conversion.

In this project, we built a calculator we can use to convert any number from any base to any other base. In number systems, “Base” tells us how many numbers you have available. Binary has 2, octal has 8, decimal has 10 etc.

The program can handle any number and base we throw at it. This program covers several aspects of programming.