

## Arquivo dv.py:

```
from math import ceil
import sys

class DigitoVerificador:

    def __init__(self, _input):
        self._cnpj = _input.upper()
        self._pesos = list()
        self.digito = 0

    def calculaAscii(self, _caracter):
        return ord(_caracter) - 48

    def calcula_soma(self):
        _tamanho_range = len(self._cnpj)
        _num_range = ceil(_tamanho_range / 8)
        for i in range(_num_range):
            self._pesos.extend(range(2,10))
        self._pesos = self._pesos[0:_tamanho_range]
        self._pesos.reverse()
        sum_of_products = sum(a*b for a, b in zip(map(self.calculaAscii, self._cnpj), self._pesos))
        return sum_of_products

    def calcula(self):

        mod_sum = self.calcula_soma() % 11

        if(mod_sum < 2):
            return 0
        else:
            return 11 - mod_sum

        return sum_of_products

if __name__ == "__main__":
    cnpj = sys.argv[1]
    dv = DigitoVerificador(cnpj)
    print(dv.calcula())
```

## Arquivo cnpj.py:

```
from dv import DigitoVerificador
import sys
import re

class CNPJ:
    def __init__(self, _input_cnpj):
        try:
            _cnpj_valido = self.__valida_formato(_input_cnpj)
            if(_cnpj_valido):
                self.cnpj = self.__remove_pontuacao(_input_cnpj)
            else:
                raise Exception("CNPJ não está no padrão aa.aaa.aaa/aaaa-dd (Para validação) ou aa.aaa.aaa/aaaa (Para geração do DV)")
        except Exception as _e:
            print(_e)
            sys.exit(0)

    def __remove_digitos_cnpj(self):
        if len(self.cnpj) == 14:
            self.cnpj_sem_dv = self.cnpj[0:-2]
        elif len(self.cnpj) == 12:
            self.cnpj_sem_dv = self.cnpj
        else:
            raise Exception("CNPJ com tamanho inválido!")

    def __remove_pontuacao(self, _input):
        return ''.join( x for x in _input if x not in "./-")

    def valida(self):
        self.__remove_digitos_cnpj()
        _dv = self.gera_dv()
        return "%s%s" % (self.cnpj_sem_dv, _dv) == self.cnpj

    def gera_dv(self):
        self.__remove_digitos_cnpj()
        dv1 = DigitoVerificador(self.cnpj_sem_dv)
        dv1char = '{}'.format(dv1.calcula())
        dv2 = DigitoVerificador(self.cnpj_sem_dv + dv1char)
        dv2char = '{}'.format(dv2.calcula())

        return "%s%s" % (dv1char,dv2char)

    def __valida_formato(self, _cnpj):
        return re.match(r'^(([A-Z]|\d){2}\.([A-Z]|\d){3}\.([A-Z]|\d){3}/([A-Z]|\d){4}(\-\d{2})?$', _cnpj)

if __name__ == "__main__":
    try:
        if len(sys.argv) < 2:
            raise Exception("Formato inválido do CNPJ.")
        _exec = sys.argv[1].upper()
        _input = sys.argv[2]
        cnpj = CNPJ(_input)
        if _exec == '-V':
            print(cnpj.valida())
        elif _exec == '-DV':
            print(cnpj.gera_dv())
        else:
            raise Exception("Opção inválida passada, as válidas são: -v para validar, -dv para gerar digito validador.")
        sys.exit()
    except Exception as _e:
        print(_e)
        sys.exit(0)
```