4.2 Ejercicio de programación 1

Primer ejercicio:
Para el primer ejercicio primero tenemos que leer el archivo, calcular las estadísticas (media, mediana, moda, desviación estándar), tener manejo de errores como datos no válidos debe marcar un error, imprimir y guardar los resultados en un archivo llamado StatisticsResults.txt, medir el tiempo de ejecución y por último cumplir con PEP8.

Cálculo:
Media: Se calcula sumando todos los números y dividiéndolo entre la cantidad de números.
Mediana: Para obtenerlo primero se deben de ordenar los datos para después ver cuál es el número del centro.
Moda: Es el número que aparece más veces en el archivo, si existen más números con la misma frecuencia todos se tomaran como moda.
Desviación estándar y varianza: La primera mide la dispersión de los datos según la media y la segunda es el cuadrado de la desviación estándar.

Resultados del archivo TC1:
Como observamos hace un correcto manejo de errores.

```
C:\Users\secdh\Desktop\New folder>python computeStatistics.py TC1.txt
Advertencia: '405s' no es un número válido.
Media: 241.91228070175438
Mediana: 239.0
Moda: [170.0, 393.0]
Varianza: 21086.305588532734
Desviación estándar: 145.211244704164447
Tiempo de ejecución: 0.0102 segundos
```

Resultados del archivo TC2:

```
C:\Users\secdh\Desktop\New folder>python computeStatistics.py TC2.txt
Media: 250.7840161861406
Mediana: 247.0
Moda: [230.0]
Varianza: 20785.369132479253
Desviación estándar: 144.17131868884064
Tiempo de ejecución: 0.0030 segundos

C:\Users\secdh\Desktop\New folder>
```

Resultados del archivo TC3:

```
C:\Users\secdh\Desktop\New folder>python computeStatistics.py TC3.txt
Media: 249.77621989860583
Mediana: 249.0
Moda: [94.0]
Varianza: 21117.277473163307
Desviación estándar: 145.3178498091797
Tiempo de ejecución: 0.0075 segundos

C:\Users\secdh\Desktop\New folder>
```

Resultados del archivo TC4:

```
C:\Users\secdh\Desktop\New folder>python computeStatistics.py TC4.txt
Media: 149.002673479087 46
Mediana: 147.75
Moda: [123.75]
Varianza: 17007.92084301886
Desviación estándar: 130.41441961308902
Tiempo de ejecución: 0.0080 segundos

C:\Users\secdh\Desktop\New folder>
```

Resultados del archivo TC5:

```
C:\Users\secdh\Desktop\New folder>python computeStatistics.py TC5.txt
Advertencia: 'ABA' no es un número válido.
Advertencia: '23,45' no es un número válido.
Advertencia: '11;54' no es un número válido.
Advertencia: 'll' no es un número válido.
Media: 241.49511400651465
Mediana: 241.0
Moda: [11.0, 19.0, 46.0, 56.0, 64.0, 76.0, 96.0, 166.0, 170.0, 211.0, 215.0, 268.0, 277.0, 278.0, 290.0, 368.0, 375.0, 393.0, 466.0]
Varianza: 21160.02196309775
Desviación estándar: 145.46484786056646
Tiempo de ejecución: 0.0020 segundos

C:\Users\secdh\Desktop\New folder>
```

Resultados del archivo TC6:

Al observar tantos números en La moda podemos concluir que probablemente muchos o todos los números de ese archivo son únicos.

```
C:\Users\secdh\Desktop\New folder>python computeStatistics.py TC6.txt
Media: 1.8790659927977443e+20
Mediana: 1.88008049965543e+20
Moda: [3.8432222947123 9e+16, 4.42636820207897e+16, 1.27923088054593e+17, 4.18459407226568e+17, 4.3346794668984 4e+17, 4.74156524686894e+17, 5.66569032951228e+17, 6.14093326870374e+17, 7.028295706280 61e+17, 8.44951320360054e+17, 9.6231693617084e+17, 1.564642285568 48e+18, 1.57225590428176e+18, 1.66781561469377e+18, 1.74876575855605e+18, 1.77016978541637e+18, 1.92240873536675e+18, 1.92718302371046e+18, 2.02185490848343e+18, 2.3609613090866e+18, 2.64578431210336e+18, 2.69468836271228e+18, 3.21493248939809e+18, 3.24658491100538e+18, 3.33406166852301e+18, 3.39523062029604e+18, 3.40869426161607e+18, 3.5143146521594e+18, 3.78753856660849e+18, 3.81523503242466e+18, 3.81537018014463e+18, 4.0698555324177e+18, 4.27494577369683e+18, 4.52034977157334e+18, 4.54055877498055e+18, 4.54876650075513e+18, 4.83445078978081e+18, 4.99002969644931e+18, 5.20030354593794e+18, 5.24675292616446e+18, 5.53638846778075e+18, 5.61356832029307e+18, 5.68608923781588e+18, 5.71084135258412e+18, 5.98536474334901e+18, 6.01426147179743e+18, 6.32564503915044e+18, 6.50108768025144e+18, 6.5325665694176e+18, 6.61522935154001e+18, 6.75907056963061e+18, 6.89061484896175e+18, 6.97349701612308e+18, 7.02188915553612e+18, 7.09596579855211e+18, 7.23819347430577e+18, 7.73096152533541e+18, 8.03614117301755e+18, 8.19914393048524e+18, 8.20606299602167e+18, 8.29364376351623e+18, 8.61220328442165e+18, 8.6693567830404e+18, 8.89145987273911e+18, 8.94158683122816e+18, 9.2877787836732e+18, 9.68469473381034e+18, 9.70122888913001e+18, 9.77849712685869e+18, 1.01589610021601e+19, 1.03072709808357e+19, 1.03318258330764e+19, 1.03946029628707e+19, 1.04740960740725e+19, 1.05161435651538e+19, 1.06448586177779e+19, 1.08597832194179e+19, 1.08991819884075e+19, 1.09969369438776e+19, 1.10724306234 5e+19, 1.12887103739089e+19, 1.14681234250897e+19, 1.16267402795968e+19, 1.16473836024915e+19, 1.17515601516222e+19, 1.19919372267645e+19, 1.20207665536304e+19, 1.20837349546841e+19, 1.21626475153919e+19, 1.23159649609944e+19, 1.24679099931811e+19, 1.24925920388305e+19, 1.25826809201157e+19, 1.26465284192004e+19, 1.27046886582263e+19, 1.27244694516694e+19, 1.28548240914957e+19, 1.28594888492724e+19, 1.29519624023452e+19, 1.32175276978713e+19, 1.34955247977497e+19, 1.35117797192946e+19, 1.37501263425331e+19, 1.37552316146802e+19, 1.38604477772675e+19, 1.40931675829438e+19, 1.41949957509623e+19, 1.42279886818961e+19, 1.44640024539755e+19, 1.44921834654226e+19, 1.46417191906504e+19, 1.48530302983184e+19, 1.49248166522665e+19, 1.50439883253 7e+19, 1.5279107293709e+19, 1.56004487201178e+19, 1.5616930326026e+19, 1.5778248738235e+19, 1.5779494332847 4e+19, 1.58867475139459e+19, 1.59858350464331e+19, 1.60313359303203e+19, 1.60521829767258e+19, 1.61139220922712e+19, 1.61820818516477e+19, 1.62126723947442e+19, 1.62803409830144e+19, 1.63354445539973e+19, 1.63706767867332e+19, 1.68346318459167e+19, 1.68411293409165e+19, 1.69086964994514e+19, 1.70276318242178e+19, 1.71934192861978e+19, 1.73390548629947e+19, 1.76455176896623e+19, 1.7663336968667e+19, 1.79755853515002e+19, 1.800207867
```

```
0063e+20, 3.744024212165 85e+20, 3.744097220951 82e+20
062e+20, 3.748464621743 95e+20]
Varianza: 1.1530904699530652e+40
Desviación estándar: 1.07382050173810 02e+20
Tiempo de ejecución: 0.0110 segundos
```

Resultados del archivo TC7:

```
C:\Users\secdh\Desktop\New folder>python computeStatistics.py TC7.txt
Advertencia: 'ABBA' no es un número válido.
Advertencia: 'ERROR' no es un número válido.
Media: 2.4746739549971625e+20
Mediana: 2.4664097307429e+20
Moda: [9596336322581500.0, 2.88418752936725e+16, 9.10526151920998e+16, 1.08852321352715e+17, 1.20138430722061e+17, 1.26200452036207e
+17, 1.62304251408896e+17, 1.95227046777269e+17, 2.48341254759032e+17, 3.00299823761729e+17, 3.59688111427003e+17, 3.65477788842383e
+17, 3.71141705319322e+17, 3.8390961035728e+17, 4.041162464747 95e+17, 4.075813487920 41e+17, 4.205193386395 13e+17, 4.513976885607 84e+
17, 4.962828450046 92e+17, 5.358234614051 83e+17, 5.719378804812 1e+17, 5.945072484075 39e+17, 6.140702766063 16e+17, 6.380343164115 73e+1
7, 6.933128315792 51e+17, 7.050080727656 34e+17, 7.298024857025 91e+17, 7.412544089839 27e+17, 7.529386361826 43e+17, 8.798031609030 25e+1
7, 8.953878536834 27e+17, 1.013670638173 98e+18, 1.067717570900 35e+18, 1.118171546369 4e+18, 1.145925855435 55e+18, 1.157942899973 29e+18
, 1.160943966422 3e+18, 1.372557196143 38e+18, 1.461467076884 88e+18, 1.463133525436 7e+18, 1.476389651444 92e+18, 1.512879123826 14e+18,
1.529958215212 51e+18, 1.561292403391 98e+18, 1.568342830162 83e+18, 1.634677632243 29e+18, 1.684831032446 21e+18, 1.772240848533 72e+18,
1.778594973821 66e+18, 1.788201731515 67e+18, 1.803351256502 44e+18, 1.844367642861 67e+18, 1.869861680534 27e+18, 1.885733551680 37e+18,
1.933523514897 34e+18, 1.977285502619 94e+18, 2.038698394809 75e+18, 2.075310976264 68e+18, 2.082208929532 43e+18, 2.133809823037 85e+18,
```

```
3536756857e+20, 4.995527375049 37e+20, 4.996628044463 09e+20, 4.996849638882 72e+20, 4.997369853672 78e+20, 4.997706899330 39e+20, 4.9977
298349933e+20, 4.997863039078 45e+20, 4.998104985482 06e+20, 4.998527087900 41e+20, 4.998692162372 45e+20, 4.999247211812 29e+20, 4.99979
529854863e+20, 4.999818609012 99e+20, 4.999940824971 51e+20]
Varianza: 2.0910793147136 53e+40
Desviación estándar: 1.4460564700984 72e+20
Tiempo de ejecución: 0.0441 segundos

C:\Users\secdh\Desktop\New folder>
```

Segundo ejercicio

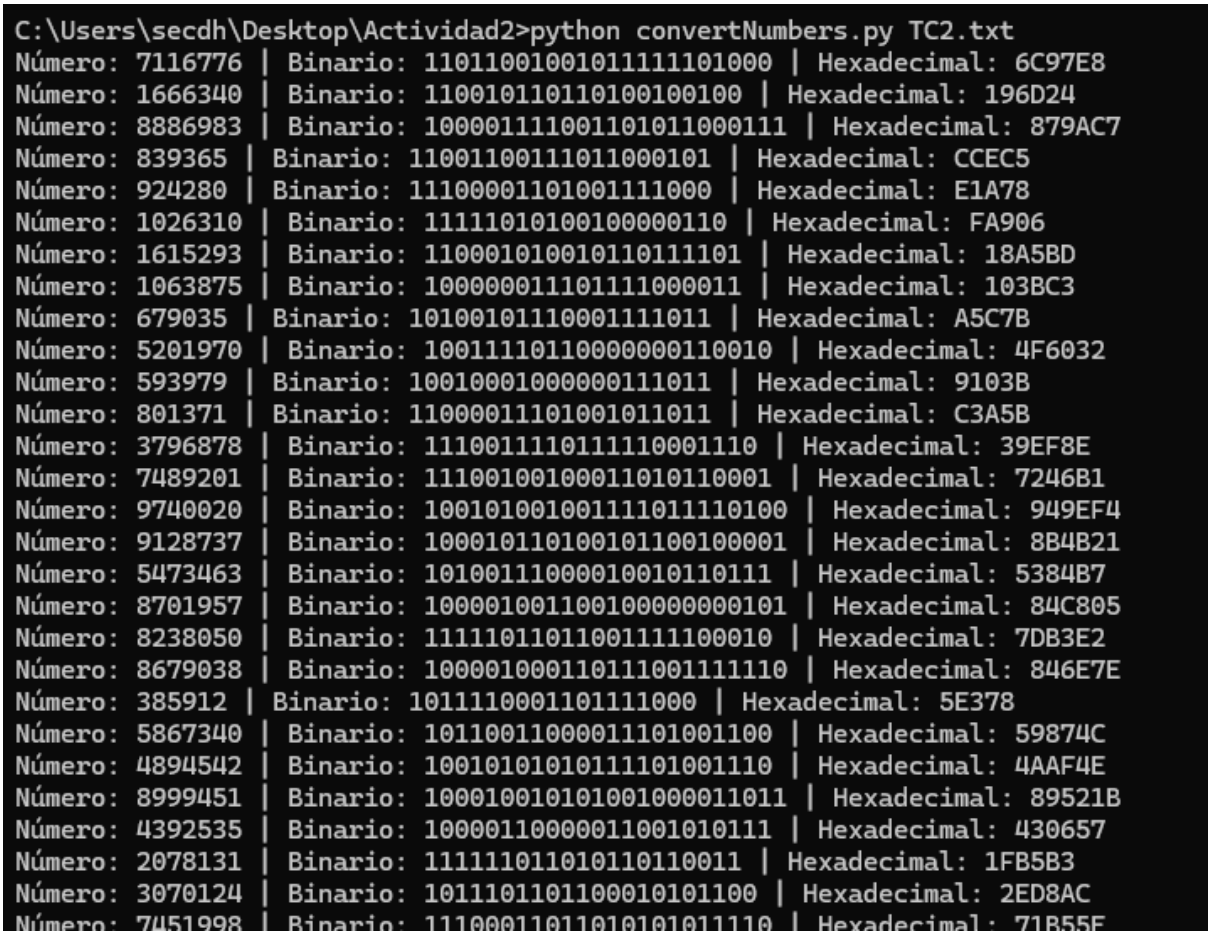
Lo que se debe hacer para este ejercicio es leer el archivo de entrada, convertir a binario y hexadecimal, guardar los resultados en un archivo txt, tener un manejo de errores y medir el tiempo de ejecución.

Resultados del archivo TC1:

```
C:\Users\secdh>cd C:\Users\secdh\Desktop\Actividad2

C:\Users\secdh\Desktop\Actividad2>python convertNumbers.py TC1.txt
Número: 6980368 | Binario: 11010101000001100010000 | Hexadecimal: 6A8310
Número: 5517055 | Binario: 10101000010111011111111 | Hexadecimal: 542EFF
Número: 1336159 | Binario: 10100011000110101111 | Hexadecimal: 14635F
Número: 6750185 | Binario: 11001101111111111101001 | Hexadecimal: 66FFE9
Número: 1771937 | Binario: 11011000010011010001 | Hexadecimal: 1B09A1
Número: 360952 | Binario: 1011000000111111000 | Hexadecimal: 581F8
Número: 5672561 | Binario: 10101101000111001110001 | Hexadecimal: 568E71
Número: 916583 | Binario: 11011111110001100111 | Hexadecimal: DFC67
Número: 2700138 | Binario: 10100100110011011010 | Hexadecimal: 29336A
Número: 9645053 | Binario: 10010011001010111111101 | Hexadecimal: 932BFD
Número: 1181110 | Binario: 10010000001011011010 | Hexadecimal: 1205B6
Número: 1492185 | Binario: 10110110001001101001 | Hexadecimal: 16C4D9
Número: 4018595 | Binario: 11110101010001101000011 | Hexadecimal: 3D51A3
Número: 7654888 | Binario: 11101001100110111101000 | Hexadecimal: 74CDE8
Número: 7062453 | Binario: 11010111100001110110101 | Hexadecimal: 6BC3B5
Número: 2478010 | Binario: 10010111001111101111010 | Hexadecimal: 25CFBA
Número: 6134768 | Binario: 10111011001101111110000 | Hexadecimal: 5D9BF0
Número: 8420417 | Binario: 10000000011111000100000 1 | Hexadecimal: 807C41
Número: 2917489 | Binario: 10110010000100011100 01 | Hexadecimal: 2C8471
Número: 3340773 | Binario: 11001011111001111001 01 | Hexadecimal: 32F9E5
Número: 1115956 | Binario: 10001000001110011010 0 | Hexadecimal: 110734
```

Resultados del archivo TC2:



Resultados del archivo TC3:

```
C:\Users\secdh\Desktop\Actividad2>python convertNumbers.py TC3.txt
Número: -39 | Binario: -100111 | Hexadecimal: -27
Número: -36 | Binario: -100100 | Hexadecimal: -24
Número: 8 | Binario: 1000 | Hexadecimal: 8
Número: 34 | Binario: 100010 | Hexadecimal: 22
Número: 17 | Binario: 10001 | Hexadecimal: 11
Número: 49 | Binario: 110001 | Hexadecimal: 31
Número: 5 | Binario: 101 | Hexadecimal: 5
Número: 39 | Binario: 100111 | Hexadecimal: 27
Número: 0 | Binario: 0 | Hexadecimal: 0
Número: 33 | Binario: 100001 | Hexadecimal: 21
Número: 12 | Binario: 1100 | Hexadecimal: C
Número: -6 | Binario: -110 | Hexadecimal: -6
Número: 27 | Binario: 11011 | Hexadecimal: 1B
Número: -4 | Binario: -100 | Hexadecimal: -4
Número: -38 | Binario: -100110 | Hexadecimal: -26
Número: 26 | Binario: 11010 | Hexadecimal: 1A
Número: 49 | Binario: 110001 | Hexadecimal: 31
Número: 29 | Binario: 11101 | Hexadecimal: 1D
Número: 42 | Binario: 101010 | Hexadecimal: 2A
Número: -16 | Binario: -10000 | Hexadecimal: -10
Número: -28 | Binario: -11100 | Hexadecimal: -1C
Número: 34 | Binario: 100010 | Hexadecimal: 22
Número: 20 | Binario: 10100 | Hexadecimal: 14
Número: 0 | Binario: 0 | Hexadecimal: 0
Número: 25 | Binario: 11001 | Hexadecimal: 19
Número: 45 | Binario: 101101 | Hexadecimal: 2D
Número: 3 | Binario: 11 | Hexadecimal: 3
```

Resultados del archivo TC4:

```
C:\Users\secdh\Desktop\Actividad2>python convertNumbers.py TC4.txt
Advertencia: 'ABC' no es un número válido.
Advertencia: 'ERR' no es un número válido.
Advertencia: 'VAL' no es un número válido.
Número: -39 | Binario: -100111 | Hexadecimal: -27
Número: -36 | Binario: -100100 | Hexadecimal: -24
Número: 8 | Binario: 1000 | Hexadecimal: 8
Número: 34 | Binario: 100010 | Hexadecimal: 22
Número: 17 | Binario: 10001 | Hexadecimal: 11
Número: 49 | Binario: 110001 | Hexadecimal: 31
Número: 5 | Binario: 101 | Hexadecimal: 5
Número: 0 | Binario: 0 | Hexadecimal: 0
Número: 33 | Binario: 100001 | Hexadecimal: 21
Número: 12 | Binario: 1100 | Hexadecimal: C
Número: -6 | Binario: -110 | Hexadecimal: -6
Número: 27 | Binario: 11011 | Hexadecimal: 1B
Número: -4 | Binario: -100 | Hexadecimal: -4
Número: -38 | Binario: -100110 | Hexadecimal: -26
Número: 26 | Binario: 11010 | Hexadecimal: 1A
Número: 49 | Binario: 110001 | Hexadecimal: 31
Número: 29 | Binario: 11101 | Hexadecimal: 1D
Número: 42 | Binario: 101010 | Hexadecimal: 2A
Número: -16 | Binario: -10000 | Hexadecimal: -10
Número: 34 | Binario: 100010 | Hexadecimal: 22
Número: 20 | Binario: 10100 | Hexadecimal: 14
```

Tercer ejercicio

En este ejercicio tenemos que leer un archivo de texto, contar la frecuencia de cada palabra del archivo, esto son ninguna biblioteca, manejar errores, guardar los resultados en un archivo txt y medir el tiempo de ejecución.

Resultados del archivo TC1:

```
C:\Users\secdh\Desktop\Actividad3>python wordCount.py TC1.txt
Resultados:
achievement: 1
adequate: 1
adventures: 1
anal: 1
andrews: 1
assessed: 1
bedding: 1
blues: 1
buying: 1
cartridge: 1
cgi: 1
championship: 1
clear: 1
club: 1
coastal: 1
collect: 1
comm: 1
confirm: 1
conservative: 2
consistent: 1
contamination: 1
could: 1
cove: 1
craps: 1
customized: 1
danish: 1
```

Resultados del archivo TC2:

```
C:\Users\secdh\Desktop\Actividad3>python wordCount.py TC2.txt
Resultados:
advantages: 1
afternoon: 1
algebra: 1
amongst: 4
amounts: 1
anchor: 1
answering: 1
apache: 1
atomic: 1
attending: 1
automation: 1
bases: 1
biz: 1
blowjob: 1
boots: 1
brass: 4
builders: 1
cannon: 1
catalog: 1
chain: 4
cheese: 1
chosen: 1
clan: 1
classes: 1
come: 1
comic: 1
compatibility: 1
```

Resultados del archivo TC3:

```
C:\Users\secdh\Desktop\Actividad3>python wordCount.py TC3.txt
Resultados:
acquisition: 1
advances: 1
affects: 1
aids: 1
allergy: 1
ambient: 1
an: 1
analyzed: 1
antiques: 1
apple: 1
archive: 1
archived: 1
argued: 1
aruba: 1
aside: 1
assembled: 1
aw: 1
ban: 1
bangbus: 1
basin: 1
bedrooms: 1
beds: 1
belt: 1
benchmark: 1
bestiality: 1
beverly: 1
bible: 1
```

Resultados del archivo TC4:

```
C:\Users\secdh\Desktop\Actividad3>python wordCount.py TC4.txt
Resultados:
adjustable: 1
admin: 1
adolescent: 1
albuquerque: 1
alternatives: 1
amazon: 1
analyst: 1
annual: 2
appreciate: 1
approve: 1
ar: 1
arabia: 1
architects: 1
arthritis: 1
asian: 1
assessed: 1
assigned: 1
ata: 1
ate: 1
attention: 1
audit: 1
australian: 1
az: 1
baby: 1
bacterial: 1
banking: 1
barrel: 1
barrier: 1
```

Resultados del archivo TC5:

```
C:\Users\secdh\Desktop\Actividad3>python wordCount.py TC5.txt
Resultados:
acquired: 1
adjust: 1
advantage: 1
affairs: 1
afterwards: 1
agenda: 1
aim: 1
albums: 1
allowed: 1
americans: 1
amsterdam: 1
andy: 1
anthropology: 1
antique: 1
anybody: 1
anytime: 1
anywhere: 1
appearing: 1
applied: 1
ar: 2
argue: 1
arise: 1
arkansas: 1
asin: 1
assignments: 2
assurance: 1
astrology: 1
```