Development Exercises – L1

Goal: The target of these exercises is to enable the student to practice and master programming skills during the course.

Instructions: You are required to submit some programming assignments However, it is STRONGLY recommended that you complete all of them. More practice, more mastery. Required exercises are denoted with **. You must upload your source code and screen images for the tests execution as evidence of your work.

Programming Exercise	Description	Test Cases
1	Ask the user for a number. Depending on whether the number is even or odd, print out an appropriate message to the user. Hint: how does an even / odd number react differently when divided by 2? If the number is a multiple of 4, print out a different message. Ask the user for two numbers: one number to check (call it num) and one number to divide by (check). If check divides evenly into num, tell that to the user. If not, print a different appropriate message.	1, 5, 4, 8, 2500, 45000, -1, p, t, 45658345213
2	Generate a random number between 1 and 9 (including 1 and 9). Ask the user to guess the number, then tell them whether they guessed too low, too high, or exactly right. (Hint: remember to use the user input lessons from the very first exercise) Keep the game going until the user types “exit” Keep track of how many guesses the user has taken, and when the game ends, print this out.	Record three runs
3	Write a program (function!) that takes a list and returns a new list that contains all the elements of the first list minus all the duplicates. Write two different functions to do this - one using a loop and constructing a list, and another using sets.	[1,1,2,3,4,5,1,5,7,7,8,9,0,0,0,0] [a,1,b,0,b,1,e,r,t,p,q] [] Add more test cases for large lists
4	Write a function that computes the standard deviation for a set of numbers coming from a list. Do not use any math module, compute the algorithm	See lists Below

5	Write a function that receives as parameters how many Fibonnaci numbers to generate and then generates them. Take this opportunity to think about how you can use functions. Make sure to ask the user to enter the number of numbers in the sequence to generate.(Hint: The Fibonnaci sequence is a sequence of numbers where the next number in the sequence is the sum of the previous two numbers in the sequence. The sequence looks like this: 1, 1, 2, 3, 5, 8, 13, …)	**0,0.5,1,3,8,2000,450000,-1,p,[]**
6	Write a function that evaluates if a given list satisfy Fibonacci sequence returning true or false if the list satisfy the criteria	**Use outputs from exercise 5**
7	Write a password generator function in Python. Be creative with how you generate passwords - strong passwords have a mix of lowercase letters, uppercase letters, numbers, and symbols. The passwords should be random, generating a new password every time the user asks for a new password.	
8**	**Write a module containing different function that computes the** **1. Sample mean** **2. Sample standard deviation** **3. Median** **4. A function that returns the n-quartil** **5. A function that returns the n-percentil**	
9**	Write a function that converts a decimal number into a Roman format	

Data set 1

3950	4930	739	4930	373	7434	6654	9275	7999	1114	1353	585	1859	9027	9399	8952
4252	8361	1741	7831	491	2320	2972	9583	7809	4045	6968	6758	8834	2333	4855	2843
9515	6881	8480	8977	8366	5145	1819	4147	6112	3973	4925	3426	1466	7685	7704	1083
1128	4207	8374	2675	9559	8350	6619	9583	1424	9419	649	7044	5199	3196	3188	3860
8251	4676	4218	3972	6297	779	75	3336	7410	6176	9910	126	2362	6381	6203	6965
4515	7667	4398	7251	6869	6288	424	5016	1653	7099	4306	6474	6594	183	8525	5651
9279	2219	1722	5194	8828	6103	2449	2176	1306	5133	3341	6786	7645	3386	1602	4716
1884	4068	4650	3527	7326	7378	4399	5210	8881	9131	9638	673	1448	3324	4616	2265
6163	3360	2029	8855	8024	4789	3025	1908	9179	7954	1038	7953	1926	2260	283	4192
4543	4892	5730	5089	7657	4505	9487	6555	3869	587	6359	1497	2046	2537	346	5575
2606	8396	2463	1559	7299	9487	8082	6087	4076	9967	8344	7443	4128	6664	3713	7399
2031	2751	3059	9281	3963	8600	182	2229	9411	3353	2781	8042	7694	9855	2188	5743
9880	9748	4195	2226	914	2220	7057	2047	6750	7667	8893	9418	208	3781	4660	2481
233	8130	4294	2465	7993	363	6056	4485	2666	4827	3184	9500	1159	7463	674	161
9711	3409	8915	1815	2241	6760	2567	9919	1200	3863	9531	2376	8939	1296	8093	7630
6908	2593	7883	5787	6164	6108	6168	490	8037	551	173	6337	7019	3124	7438	3414
8832	6330	4653	8470	55	861	8062	1125	532	4672	6992	5047	4803	4237	5125	8410
5086	6981	2736	9832	8907	8390	8887	3478	9619	7867	5871	6551	3409	5142	3027	4814
7888	4929	5439	5897	8839	6750	700	3336	8915	4630	1052	3129	4728	5357	1953	9044
4069	8767	6408	9503	7279	3268	6502	7852	9882	4355	1854	9263	3829	9163	4903	4946
4619	2937	2336	1860	4307	4694	2075	3979	8730	9828	4470	513	6339	4039	5423	8046
5392	2378	641	6568	9863	8749	16	2124	5315	7796	9953	8324	5990	8682	2815	3662
4092	8870	1745	5883	4030	5804	2611	6334	955	3868	5832	5675	9394	9289	2797	7263
7528	9724	6577	4131	3315	130	8246	9478	8052	5213	3174	683	3056	1288	5810	964
7557	3599	8442	4854	4999	1156	1030	3445	8575	2315	1858	7700	254	1879	9541	5753
5950	7027	2167	6631	8057	6588	5259	3353	4781	8861	5881	1657	8036	6074	3734	5144
755	6180	8261	1960	2732	2635	4326	9449	7656	1485	9544	8384	8387	6822	5683	5567
7878	4944	3979	908	1868	9712	5052	5612	2226	9147	7755	2881	8102	5767	4351	5852
9219	7164	5969	6224	3698	711	9984	6133	147	2717	3601	8129	2976	6102	6476	6510
9505	285	4980	8307	7785	1770	3951	5847	8757	653	49	6587	7691	8124	6733	1342

5455	6533	4828	9218	6902	392	4313	2561	8200	8532	3822	1749	5075	2725	7315	1596
8871	9665	7185	4573	7907	8068	1618	1556	4448	506	1822	6550	6042	2996	9655	4172
6032	5109	2244	4545	9881	278	2104	2834	7389	2874	1024	9458	475	7102	9816	810
398	1076	4711	5696	1677	4307	528	1209	6108	4300	8680	214	5271	9228	1088	8073
8887	5700	7739	7462	8614	2153	6696	7461	295	765	8949	9105	5114	5924	7439	8095
4428	6763	6697	8604	4791	886	8568	3988	9172	9484	4296	3260	9413	401	2888	5691
4928	6565	3357	5934	7144	1415	5110	2840	3694	7064	8125	125	4709	2584	2647	5495
7137	4998	8501	4363	5881	5875	2750	9	6371	550	1652	5809	7330	4711	5924	659
6579	3585	2864	6695	9030	4871	8332	3376	7723	3909	1071	5159	6382	5256	8218	5092
4431	6415	3920	2604	1322	2824	826	3325	5190	1193	6000	5682	1311	7926	9425	7399

Data set 2

1880	9914	5528	4723	2391	4551	8623	9827	3084	618	4176	3621	5736	7045	7208	2281
7243	5924	8083	3234	6329	9100	6648	5793	118	6587	1602	2671	7934	9599	7470	3952
4655	5911	4960	4094	7628	2197	7539	9677	5980	8893	3996	8506	2394	3878	3570	5102
2592	4216	6094	6428	8218	6442	22	1239	1329	1890	5971	5005	3017	6068	3793	8516
1704	8612	1799	7570	8573	2596	3261	5036	3084	8389	3417	5529	3176	8680	4820	9513
6956	9925	8161	6512	4462	4993	6835	7589	3744	8090	9415	7266	7304	8924	6049	1400
974	5351	2888	2072	1968	2228	643	4688	8559	9520	1281	6468	7903	1188	8281	250
3227	9948	3093	6592	6108	389	404	675	9485	8663	2220	8341	7233	7143	4691	8515
7685	7557	7505	8193	5845	6418	6567	4749	2025	2056	3283	9230	4739	4247	256	7381
8409	3754	7497	9650	2157	6929	7029	2241	8421	5755	3265	4580	6164	827	8763	5487
6815	1710	3179	8233	456	6374	8124	9118	2660	1950	6585	7101	8697	3922	8850	9136
9292	8734	7146	1968	2688	4698	4299	258	9309	6497	8184	8162	6389	2567	9396	5664
7143	7457	4676	6819	6042	4392	1769	1691	2078	2499	4807	2259	1428	5162	1951	1337