Introduction to Programming in Python

Assignment 3 (Mozart Waltz Generator) Discussion

reverse.py

Standard input	a sequence of strings
Standard output	the strings in reverse order

>_ ~/workspace/mozart_waltz_generator

```
$ python3 reverse.py
b o l t o n
<ctrl-d>
n o t l o b
$ python3 reverse.py
m a d a m
<ctrl-d>
m a d a m
```

Read all strings from standard input into a list a (use `stdio.readAllStrings()`)

Set n to the size of a

For each $i \in [0, n/2)$
- Exchange $a[i]$ with $a[n - i - 1]$

For each $i \in [0, n)$
- If $i < n - 1$, write $a[i]$ with a space after; otherwise, write $a[i]$ with a newline after

distance.py

Command-line input	n (int)
Standard input	two size-n lists x and y of floats
Standard output	the Euclidean distance between the two vectors represented by x and y

~/workspace/mozart_waltz_generator

```
$ python3 distance.py 2
1 0 <enter>
0 1 <enter>
1.4142135623730951
$ python3 distance.py 5
-9 1 10 -1 1 <enter>
-5 9 6 7 4 <enter>
13.0
```

Accept n (int) as command-line argument

Create an empty list x

For each $i \in [0, n)$
- Append to x a float read from standard input (use `stdio.readFloat()`)

Create a list y of floats similar to x

Set *distance* to 0.0

For each $i \in [0, n)$
- Increment *distance* by $(x[i] - y[i])^2$

Write $\sqrt{distance}$

birthday.py

Command-line input	*trials* (int)
Standard output	average number of individuals that must be sampled until there is a match in their birthdays

~/workspace/mozart_waltz_generator

```
$ python3 birthday.py 1000
24
$ python3 birthday.py 1000
25
```

Set `DAYS_PER_YEAR` to 365

Accept *trials* (int) as command-line argument

Set *count* to 0

For each t in $[0, \textit{trials})$
- Set *birthdaysSeen* to a list of size `DAYS_PER_YEAR` with all elements set to `False` (use `stdarray.create1D()`)
- Repeat forever
    - Increment *count* by 1
    - Set *birthday* to a random int from $[0, \texttt{DAYS\_PER\_YEAR})$
    - If *birthday* was seen before, break; otherwise, record that we are seeing it now

Write $\textit{count}/\textit{trials}$ (use integer division)

transpose.py

Command-line input	m (int) and n (int)
Standard input	$m \times n$ floats representing the elements of an $m \times n$ matrix a
Standard output	the transpose of a

~/workspace/mozart_waltz_generator

```
$ python3 transpose.py 2 2
1 2 <enter>
3 4 <enter>
1.0 3.0
2.0 4.0
$ python3 transpose.py 2 3
1 2 3 <enter>
4 5 6 <enter>
1.0 4.0
2.0 5.0
3.0 6.0
```

Accept m (int) and n (int) as command-line arguments

Create an $m \times n$ list a with all elements set to `None` (use `stdarray.create2D()`)

For each $i \in [0, m)$
- For each $j \in [0, n)$
  - Set $a[i][j]$ to a float read from standard input (use `stdio.readFloat()`)

Create an $n \times m$ list c with all elements set to `None` (use `stdarray.create2D()`)

For each $i \in [0, n)$
- For each $j \in [0, m)$
  - Set $c[i][j]$ to $a[j][i]$

For each $i \in [0, n)$
- For each $j \in [0, m)$
  - If $j < m - 1$, write $c[i][j]$ with a space after; otherwise, write $c[i][j]$ with a newline after

pascal.py

Command-line input	n (int)
Standard output	Pascal's triangle $\mathcal{P}_n$

>_ ~/workspace/mozart_waltz_generator

```
$ python3 pascal.py 5
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
```

Accept n (int) as command-line argument

Create a list a of size $n + 1$ with all elements set to `None` (use `stdarray.create1D()`)

For each $i \in [0, n]$
- Set $a[i]$ to a list of size $i + 1$ with all elements set to 1 (use `stdarray.create1D()`)

For each $i \in [0, n]$
- For each $j \in [1, i)$
  - Set $a[i][j]$ to $a[i-1][j-1] + a[i-1][j]$

For each $i \in [0, n]$
- For each $j \in [0, i]$
  - If $j < i$, write $a[i][j]$ with a space after; otherwise, write $a[i][j]$ with a newline after

A waltz consists of two parts, the minuet and the trio, each comprised of 16 measures

The file `data/mozart.wav` provides an example of a waltz — play it manually to get an idea of what a waltz sounds like

There are 176 possible minuet measures (labeled $1, 2, \ldots 176$) and 96 possible trio measures (labeled $1, 2, \ldots, 96$)

Corresponding to each minuet and trio measure, there's an audio (`.wav`) file under the `data` directory that we can play

Example: `data/M167.wav` corresponds to the minuet measure 167 and `data/T42.wav` corresponds to the trio measure 42 — play these files manually to get a sense of what each measure sounds like

Goal: write a program to generate a waltz and another program to play the waltz

The first 16 minuet measures of the waltz are generated as follows

- To generate the jth measure, roll *two* fair dice (let's call the sum of the rolls i)
- Write the number in row i and column j of the following *minuet* table

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2	96	22	141	41	105	122	11	30	70	121	26	9	112	49	109	14
3	32	6	128	63	146	46	134	81	117	39	126	56	174	18	116	83
4	69	95	158	13	153	55	110	24	66	139	15	132	73	58	145	79
5	40	17	113	85	161	2	159	100	90	176	7	34	67	160	52	170
6	148	74	163	45	80	97	36	107	25	143	64	125	76	136	1	93
7	104	157	27	167	154	68	118	91	138	71	150	29	101	162	23	151
8	152	60	171	53	99	133	21	127	16	155	57	175	43	168	89	172
9	119	84	114	50	140	86	169	94	120	88	48	166	51	115	72	111
10	98	142	42	156	75	129	62	123	65	77	19	82	137	38	149	8
11	3	87	165	61	135	47	147	33	102	4	31	164	144	59	173	78
12	54	130	10	103	28	37	106	5	35	20	108	92	12	124	44	131

- For example if $j = 4$ and $i = 7$ (from die rolls 4 and 3), then write 167

The next 16 trio measures of the waltz are generated as follows

- To generate the jth measure, roll *one* fair die (let's call the roll i)
- Write the number in row i and column j of the following *trio* table

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	72	6	59	25	81	41	89	13	36	5	46	79	30	95	19	66
2	56	82	42	74	14	7	26	71	76	20	64	84	8	35	47	88
3	75	39	54	1	65	43	15	80	9	34	93	48	69	58	90	21
4	40	73	16	68	29	55	2	61	22	67	49	77	57	87	33	10
5	83	3	28	53	37	17	44	70	63	85	32	96	12	23	50	91
6	18	45	62	38	4	27	52	94	11	92	24	86	51	60	78	31

- For example if $j = 3$ and $i = 2$ (from die roll 2), then write 42

The *minuet* and *trio* tables are defined in the file `data/mozart.txt`

The first 16 minuet measures of the waltz are played as follows

- To play the ith measure, play the `.wav` file under `data` whose name starts with "M" and is followed by the number i
- For example, if $i = 167$, play the file `data/M167.wav`

The next 16 trio measures of the waltz are played as follows

- To play the ith measure, play the `.wav` file under `data` whose name starts with "T" and is followed by the number i
- For example, if $i = 42$, play the file `data/T42.wav`

generatewaltz.py

Standard input	the minuet and trio tables
Standard output	a random sequence of 32 measures according to the rules described above

~/workspace/mozart_waltz_generator

```
$ python3 generatewaltz.py < data/mozart.txt
69 95 27 103 105 129 21 24 66 155 48 34 43 18 89 78 72 39 59 68 29 7 15 94 76 34 93 77 12 95 47 10
$ python3 generatewaltz.py < data/mozart.txt
32 84 27 50 153 97 36 100 16 4 150 34 51 115 1 78 18 3 59 74 37 43 52 71 9 20 32 79 57 35 90 10
```

Part II (Mozart Waltz Generator) · Problem 6 (Generating the Waltz)

Create a 2D list called *minuetMeasures* with dimensions 11×16 (use `stdarray.create2D()`)

Populate *minuetMeasures* with values read from standard input (use `stdio.readInt()`)

Create a 2D list called *trioMeasures* with dimensions 6×16 (use `stdarray.create2D()`)

Populate *trioMeasures* with values read from standard input (use `stdio.readInt()`)

For each $j \in [0, 15]$
- Set i to the sum of two die rolls (each a random number from $[1, 6]$)
- Write $minuetMeasures[i-2][j]$ with a space after

For each $j \in [0, 15]$
- Set i to the value of a die roll (a random number from $[1, 6]$)
- Write $trioMeasures[i-1][j]$ with a space after

Write a newline

playwaltz.py

Standard input	a sequence of 32 measures of a waltz
Standard audio	the waltz

>_ ~/workspace/mozart_waltz_generator

```
$ python3 generatewaltz.py < data/mozart.txt | python3 playwaltz.py
```

Part II (Mozart Waltz Generator) · Problem 7 (Playing the Waltz)

Set *measures* to a list of ints (representing measures of a waltz) read from standard input (use `stdio.readAllInts()`)

Exit the program with the message "A waltz must contain exactly 32 measures" if *measures* does not contain 32 values (use `sys.exit()`)

Exit the program with the message "A minuet measure must be from [1, 176]" if the any of the first 16 values of *measures* is not from the interval $[1, 176]$ (use `sys.exit()`)

Exit the program with the message "A trio measure must be from [1, 96]" if the any of the last 16 values of *measures* is not from the interval $[1, 96]$ (use `sys.exit()`)

For each *v* of the first 16 values in *measures*

- Set *filename* to "data/M" + *v*
- Play the audio file with the name *filename* (use `stdaudio.playFile()`)

For each *v* of the last 16 values in *measures*

- Set *filename* to "data/T" + *v*
- Play the audio file with the name *filename* (use `stdaudio.playFile()`)