Advent of Code 2023

Day 4

In[679]:=
```
SetDirectory[NotebookDirectory[]]
```
Out[679]=
```
/home/rlupi/src/aoc2023/4
```

In[680]:=
```
input = Import["input.txt", "Lines"];
```

In[681]:=
```
Length[input]
```
Out[681]=
```
204
```

Part 1

In[682]:=
```
testInput = StringSplit["Card 1: 41 48 83 86 17 | 83 86  6 31 17  9 48 53
Card 2: 13 32 20 16 61 | 61 30 68 82 17 32 24 19
Card 3:  1 21 53 59 44 | 69 82 63 72 16 21 14  1
Card 4: 41 92 73 84 69 | 59 84 76 51 58  5 54 83
Card 5: 87 83 26 28 32 | 88 30 70 12 93 22 82 36
Card 6: 31 18 13 56 72 | 74 77 10 23 35 67 36 11", EndOfLine]
```
Out[682]=
```
{Card 1: 41 48 83 86 17 | 83 86  6 31 17  9 48 53,
Card 2: 13 32 20 16 61 | 61 30 68 82 17 32 24 19,
Card 3:  1 21 53 59 44 | 69 82 63 72 16 21 14  1,
Card 4: 41 92 73 84 69 | 59 84 76 51 58  5 54 83,
Card 5: 87 83 26 28 32 | 88 30 70 12 93 22 82 36,
Card 6: 31 18 13 56 72 | 74 77 10 23 35 67 36 11}
```

In[683]:=

```
parseScratchCards[input_] := Module[{parseNumbers},
  parseNumbers[numbers_] := Map[Interpreter[Integer], StringSplit[numbers, Whitespace]];
  First /@ StringCases[input, "Card" ~~ Whitespace ~~ card : NumberString ~~ ":" ~~
      Whitespace ~~ w : Repeated[(Whitespace) ... ~~ NumberString] ~~ " | " ~~
      n : Repeated[(Whitespace) ... ~~ NumberString] :> <|"card" → Interpreter[Integer][card],
      "win" → parseNumbers[w], "num" → parseNumbers[n]|>] // Dataset]
testData = parseScratchCards[testInput];
testData // Normal (* more concise output *)
```

Out[685]=

```
{<|card → 1, win → {41, 48, 83, 86, 17}, num → {83, 86, 6, 31, 17, 9, 48, 53}|>,
 <|card → 2, win → {13, 32, 20, 16, 61}, num → {61, 30, 68, 82, 17, 32, 24, 19}|>,
 <|card → 3, win → {1, 21, 53, 59, 44}, num → {69, 82, 63, 72, 16, 21, 14, 1}|>,
 <|card → 4, win → {41, 92, 73, 84, 69}, num → {59, 84, 76, 51, 58, 5, 54, 83}|>,
 <|card → 5, win → {87, 83, 26, 28, 32}, num → {88, 30, 70, 12, 93, 22, 82, 36}|>,
 <|card → 6, win → {31, 18, 13, 56, 72}, num → {74, 77, 10, 23, 35, 67, 36, 11}|>}
```

In[686]:=

```
points1[0] := 0
points1[n_] := 2^(n - 1)
```

In[688]:=

```
matching = Intersection[#win, #num] & /* Length;
```

In[689]:=

```
testData[All, matching /* points1]
```

Out[689]=

8
2
2
1
0
0

In[690]:=

```
testData[All, matching /* points1] // Total
```

Out[690]=

```
13
```

In[691]:=

```
inputData = parseScratchCards[input];
```

In[692]:=

```
inputData[All, matching/*points1]
```

Out[692]=

512	512	8	512	512	512	2	4	4	8	512	8
256	32	32	64	2	16	4	0	0	1	0	4
256	128	1	512	0	256	1	512	512	0	32	512
64	256	2	128	2	0	1	0	2	2	0	0
4	0	4	128	512	32	1	2	0	16	2	2
0	0	0	512	8	2	512	256	64	0	8	32
0	4	0	1	1	0	512	512	8	128	512	8
64	64	512	512	4	0	16	0	0	1	2	1
0	0	1	0	512	0	512	32	0	512	8	512
512	512	512	2	256	1	8	16	0	8	0	0
2	1	0	512	512	512	512	512	512	512	32	16
32	4	64	32	256	16	16	1	16	2	0	1
2	1	0	256	128	8	512	1	512	64	32	1
16	0	0	4	2	1	0	64	4	64	128	512
512	512	2	512	2	256	128	256	8	0	2	2
8	0	0	1	0	512	0	512	128	2	32	512
512	512	128	8	16	8	8	8	0	2	0	0

In[693]:=

```
inputData[All, matching/*points1] // Total
```

Out[693]=

27 059

Part 2

In[694]:=

```
cardsPart2 = Block[{m = matching[#]}, If[m === Null, {}, Range[#card + 1, #card + m]]] &;
```

In[695]:=
```
testData[All, cardsPart2]
```
Out[695]=

{2, 3, 4, 5}
{3, 4}
{4, 5}
{5}
{}
{}

In[696]:=
```
winningCards[points_] := Module[{updates, f},
  updates = MapIndexed[{#1, #2[[1]]} &, points];
  f[counts_, {wins_, n_}] := Fold[ReplacePart[#1, #2 → counts[[n]] + #1[[#2]]] &, counts, wins];
  Fold[f, Table[1, Length[points]], updates // Normal]
 ];
testData[All, cardsPart2] // winningCards
```
Out[697]=
```
{1, 2, 4, 8, 14, 1}
```
In[698]:=
```
testData[All, cardsPart2] // winningCards // Total
```
Out[698]=
```
30
```
In[699]:=
```
inputData[All, cardsPart2] // winningCards // Total
```
Out[699]=
```
5 744 979
```