Advent of Code 2023

Day 2

In[40]:=
```
SetDirectory[NotebookDirectory[]]
```

Out[40]=
```
/home/rlupi/src/aoc2023
```

In[41]:=
```
testInput = StringSplit["Game 1: 3 blue, 4 red; 1 red, 2 green, 6 blue; 2 green
Game 2: 1 blue, 2 green; 3 green, 4 blue, 1 red; 1 green, 1 blue
Game 3: 8 green, 6 blue, 20 red; 5 blue, 4 red, 13 green; 5 green, 1 red
Game 4: 1 green, 3 red, 6 blue; 3 green, 6 red; 3 green, 15 blue, 14 red
Game 5: 6 red, 1 blue, 3 green; 2 blue, 1 red, 2 green", EndOfLine]
```

Out[41]=
```
{Game 1: 3 blue, 4 red; 1 red, 2 green, 6 blue; 2 green,
Game 2: 1 blue, 2 green; 3 green, 4 blue, 1 red; 1 green, 1 blue,
Game 3: 8 green, 6 blue, 20 red; 5 blue, 4 red, 13 green; 5 green, 1 red,
Game 4: 1 green, 3 red, 6 blue; 3 green, 6 red; 3 green, 15 blue, 14 red,
Game 5: 6 red, 1 blue, 3 green; 2 blue, 1 red, 2 green}
```

Part 1

In[42]:=
```
ParseCubes[cubes_] :=
 Association /@ StringCases[StringSplit[cubes, "; "], n : NumberString ~~ Whitespace ~~
     color : WordCharacter .. :> <|Symbol[color] -> Interpreter[Integer][n]|>]
```

In[43]:=
```
ParseGames[input_] :=
 Catenate@StringCases[input, "Game " ~~ game : NumberString ~~ ":" ~~ (cubes__) ~~
     EndOfLine :> <|Interpreter[Integer][game] -> ParseCubes[cubes]|>]
```

In[44]:=
```
test = ParseGames[testInput]
```

Out[44]=
```
{<|1 -> {<|blue -> 3, red -> 4|>, <|red -> 1, green -> 2, blue -> 6|>, <|green -> 2|>}|>,
 <|2 -> {<|blue -> 1, green -> 2|>, <|green -> 3, blue -> 4, red -> 1|>, <|green -> 1, blue -> 1|>}|>,
 <|3 -> {<|green -> 8, blue -> 6, red -> 20|>, <|blue -> 5, red -> 4, green -> 13|>,
    <|green -> 5, red -> 1|>}|>, <|4 -> {<|green -> 1, red -> 3, blue -> 6|>,
    <|green -> 3, red -> 6|>, <|green -> 3, blue -> 15, red -> 14|>}|>,
 <|5 -> {<|red -> 6, blue -> 1, green -> 3|>, <|blue -> 2, red -> 1, green -> 2|>}|>}
```

In[45]:=
```
testPredicate = (red ≤ 12) && (green ≤ 13) && (blue ≤ 14)
```
Out[45]=
```
red ≤ 12 && green ≤ 13 && blue ≤ 14
```

In[46]:=
```
ClearAll[ValidGame, ValidTurn, ValidGameQ];
ValidTurn[draws_, pred_] := (And @@ (SatisfiableQ[pred /. #] & /@ draws));
ValidGame[turns__, pred_] := And @@ (ValidTurn[#, pred] & /@ turns);
ValidGameQ[pred_] := ValidGame[#, pred] &;
```

In[50]:=
```
SnowIsland[games_, pred_] :=
 Query[Select@ValidGameQ[pred]][games] // Keys // Total // First
```

In[51]:=
```
SnowIsland[test, testPredicate]
```
Out[51]=
```
8
```

In[52]:=
```
input = Import["d2-input.txt", "Lines"] // ParseGames;
Length[input]
```
Out[53]=
```
100
```

In[54]:=
```
SnowIsland[input, testPredicate]
```
Out[54]=
```
2795
```

Part 2

In[55]:=
```
SnowIsland2[games_] :=
 Dataset[games][All, All, Merge[Max]][All, All, Values /* Apply[Times]] // Values // Total // First
```

In[56]:=
```
SnowIsland2[test]
```
Out[56]=
```
2286
```

In[57]:=
```
SnowIsland2[input]
```
Out[57]=
```
75 561
```