948. Bag of Tokens

You are given initial values:

power : any
score : 0
Bag of tokens

Maximize the total score by strategically playing these tokens.
For each move you can play an unused token in the following ways:

- If current token >= power , power -= tokens[i] and score +=1
- If score >= 1 , power += tokens[i] , score -= 1

Return maximum score possible using any # of tokens.

Ex) Input: tokens: [100]
power: 50

Output: 0

Reason: both conditionals are wrong.

Goal:

If power >= token, find the smallest token and subtract to power ++score
If score >= 1 find the Largest token and add to power --score

If we can maximize power when subtracting score, we maximize chance of having multiple min values to increase score.

```
int   maxScore( std::vector<int> tokens , int power) {
      int score{};
      std::sort( tokens.begin(), tokens.end());
      int min    {};  int maxScore {};
      int max    ( tokens.size() - 1 );
      for( int i=0; i< tokens.size(); ++i) {   while (min <= max)
          if ( tokens[min] < power ) {
               power -= tokens[min];
               ++score;  ++min;
               maxScore = std::max(maxScore, score);
          }
          else if ( score >= 1 ) {
               power += tokens[max];
               --score;  --max;
          }
          else break;
      }
      return maxScore;
}
```