Submissions

Leaderboard

All Contests > Practice-1-SDA > Darts 501

Darts 501

Problem

Discussions

≜ locked

23 | Ø

C++

f y in

Max Score: 100 Difficulty: Medium Rate This Challenge:

Single Scoring 1 x number Submissions: 115 12 Triple Ring Inner Bullseye More 3 x number 50 Points 13 6 Outer Bullseye 8 25 Points Double Ring 2 x number/ 19 Out of Play No score

чудеха какви и колко различни стратегии имат, които водят до бърза победа. 501 е най-популярната дартс игра, в която играчите започват с 501 точки и се стремят да стигнат до 0 с възможно най-малко на брой хвърляния, като се редуват да хвърлят по 3 стрели. Всички точки, които отбелязват, се изваждат от оставащите им точки.

След успешния турнир по дартс на Freshers' Weekend 2019, от ФСС решиха да направят ежемесечен

турнир на играта 501. С новия турнир обаче, дойдоха и нови тревоги за участниците. Те постоянно се

Мишената е разделена на 20 части, отговарящи на числата от 1 до 20. Всяка от тези 20 части съдържа единичен сектор, който дава точки равни на неговата стойност, двоен сектор (дубъл), равняващ се на

мишената се намира центъра, който също е разделен на 2 части. Външният (по-голям) кръг дава 25

съответното число, умножено по 2 и троен сектор (требъл), който дава утроена стройност. В средата на

точки, а вътрешния (бул) е дубъл и дава 50. (Вижте картинката за по-подробна информация) При едно хвърляне на 3 стрели максималния брой точки, който може да се постигне е 180 (3 х 60). За да се излезе и съответно да се спечели играта трябва да се направи така наречения "чекаут". "Чекаутът" е уцелване на точно толкова точки, колкото са ти останали. Също така, за да се излезе е нужно последната стрела да уцели двоен сектор (дубъл) (вътрешният централен сектор също се брои за

дубъл). Често срещан проблем пред дартс играчите е да разберат дали и по колко начина могат да "изчистят" всичките си останали точки. От вас се иска да напишете програма, която изчислява по колко различни начина може да се направи това при оставащи **N** точки.

Два чекаута са различни ако са от различен брой хвърляния (1, 2 или 3) или ако някое от хвърлянията се различава по уцелен сектор. (приемаме, че извън мишената е отделен сектор))

Имайте предвид, че е възможно играчът да пропусне мишената (да направи 0 точки). (вижте

Повече информация за правилата на играта 501 можете да намерите тук: http://www.nicedarts.com/how_to_501 Input Format

От първия ред се въвежда числото **N** – оставащите точки.

Constraints

примерите)

 $1 \le N \le 180$

Output Format

Отпечатайте едно число - по колко начина могат да се "изчистят" оставащите точки.

Sample Output 0

10

Sample Input 0

Explanation 0 Възможните излизания са: С едно хвърляне: двойно 2

С три хвърляния: пропуск, пропуск и двойно 2; пропуск, 2 и двойно 1; пропуск, двойно 1 и двойно 1; 1, 1

и двойно 1; 2, пропуск и двойно 1; двойно 1, пропуск и двойно 1; Следователно има общо 10 различни начина.

Sample Input 1

168

С две хвърляния: пропуск и двойно 2; 2 и двойно 1; двойно 1 и двойно 1

Sample Output 1 0

Sample Input 2

50

Explanation 1

Sample Output 2

Не е възможно точките да се изчистят с 3 хвърляния.

1 ▼#include <cmath>

1371

```
2 #include <cstdio>
    #include <vector>
    #include <iostream>
    #include <algorithm>
  6 using namespace std;
  7 vint isCheckoutPossible(int points){
        int counter=0;
  9
         if(points>170||points<=0){</pre>
10 🔻
11
             return 0;
12
        for(int i=0;i<=20;i++){
13 ▼
14
15 ▼
             for(int k=0; k<=20; k++){
16
17 ▼
                 for(int l=1;l<=20;l++){
18
                     if(i+k+l*2==points){
19 ▼
20
                         counter++;
21
22 🔻
                     if(i*2+k+l*2==points &&i!=0 ){
23
                         counter++;
24
25 ▼
                      if(i+k*2+l*2==points && k!=0){
26
                         counter++;
27
28 ▼
                     if(i*2+k*2+l*2==points && i!=0 && k!=0){
29
                         counter++;
30
31 🔻
                      if(i*3+k+l*2==points && i!=0){
32
                         counter++;
33
34 ▼
                      if(i+k*3+l*2==points && k!=0){
35
                         counter++;
36
                     if(i*3+k*3+l*2==points && i!=0 && k!=0 ){
37 ▼
38
                         counter++;
39
40 ▼
                     if(i*2+k*3+l*2==points && i!=0 && k!=0 ){
41
                         counter++;
42
                     if(i*3+k*2+l*2==points && i!=0 && k!=0 ){
43 ▼
44
                         counter++;
45
46 ▼
                     if(i+k==points-50 && l==20){
47
                         counter++;
48
                     if(i*2+k==points-50 && i!=0 && l==20){
49 ▼
50
                         counter++;
51
52 ▼
                     if(i*3+k==points-50 && i!=0 && l==20){
53
                         counter++;
54
55 ▼
                     if(i+k*2==points-50 && k!=0 && l==20){
56
                         counter++;
57
58 ▼
                     if(i+k*3==points-50 && k!=0 && l==20){
59
                         counter++;
60
61 ▼
                     if(i*2+k*2==points-50 && k!=0 && i!=0 && l==20){
62
                         counter++;
63
64 ▼
                     if(i*2+k*3==points-50 && k!=0 && i!=0 && l==20){
65
                         counter++;
66
67 ▼
                     if(i*3+k*2==points-50 && k!=0 && i!=0&& l==20){
68
                         counter++;
69
70 ▼
                     if(i*3+k*3==points-50 && k!=0 && i!=0 && l==20){
71
                         counter++;
72
73
74
75
76
77
        for(int i=0;i<=20;i++){
78 ▼
79
             for(int k=1; k<=20; k++){
80 ▼
81
82 🔻
                 if(k*2+i==points){
83
                     counter++;
84
85 ▼
                 if(k*2+i*2==points && i!=0){
86
                     counter++;
87
88 🔻
                 if(k*2+i*3==points && i!=0){
89
                     counter++;
90
                 if(i==points-50 && k==20){
91 ▼
92
                     counter++;
93
94 ▼
                 if(i*2==points-50 && i!=0 && k==20){
95
                     counter++;
96
                 if(i*3==points-50 && i!=0 && k==20){
97 ▼
98
                     counter++;
99
100 ▼
                 if(i+25+k*2==points){
101
                         counter+=2;
102
                 if(i*2+25+k*2==points && i!=0){
103 ▼
104
                     counter+=2;
105
106 ▼
                 if(i*3+25+k*2==points && i!=0){
107
                     counter+=2;
108
109 ▼
                  if(i+50+k*2==points){
110
                         counter+=2;
111
                 if(i*2+50+k*2==points && i!=0){
112 ▼
113
                     counter+=2;
114
115 ▼
                 if(i*3+50+k*2==points && i!=0){
116
                     counter+=2;
117
118 ▼
                 if(i+25+50==points && k==20){
119
                         counter+=2;
120
                 if(i*2+25+50==points && i!=0 && k==20){
121 ▼
122
                     counter+=2;
123
124 ▼
                 if(i*3+25+50==points && i!=0 && k==20){
125
                     counter+=2;
126
                 if(i*2+25+25==points && i!=0 && k==20){
127 ▼
128
                     counter++;
129
130 ▼
                 if(i*2+50+50==points && i!=0 && k==20){
131
                     counter++;
132
133
134 ▼
                 if(i+50+50==points && k==20){
135
                         counter+=2;
136
137 ▼
                 if(i*2+50+50==points && i!=0 && k==20){
138
                     counter+=2;
139
140 ▼
                 if(i*3+50+50==points && i!=0 && k==20){
141
                     counter+=2;
142
143 ▼
                 if(k*2 + 25==points && i==20){
144
                     counter++;
145
146 ▼
                  if(k*2 +50==points && i==20){
147
                     counter+=2;
148
149 ▼
                 if(k*2 +25 +50==points && i==20){
150
                     counter+=2;
```

Run Code

Submit Code

Line: 1 Col: 1

Contest Calendar | Interview Prep | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy | Request a Feature

cout<<isCheckoutPossible(points);</pre>

counter++;

counter++;

if(points==100){

return counter;

int points;

return 0;

cin>>points;

if((points%2==0 && points<=40)||points==50){

151

152

153

154

155

156

158

159

160 ▼

161

162

163

165

167

168

169

170

172

171 }

164 }

166 **▼**int main() {

157 ▼