

LAPORAN TUGAS KECIL 1
IF 2211
Penyelesaian Permainan Kartu 24 dengan
Algoritma Brute Force



Ditujukan untuk memenuhi salah satu tugas kecil mata kuliah IF2211 Strategi Algoritma pada Semester II Tahun Akademik 2022/2023

Disusun oleh:

Haidar Hamda

13521105

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN
INFORMATIKA INSTITUT TEKNOLOGI
BANDUNG
2023

1. Algoritma Brute Force

Langkah-langkah penyelesaian permainan kartu 24 dengan algoritma brute force yang telah dibuat sebagai berikut:

1. Bentuk ekspresi matematika dengan penulisan infix.
2. Ubah ekspresi infix menjadi ekspresi postfix.
3. Hitung ekspresi postfix.
4. Simpan ekspresi jika hasil perhitungan bernilai 24.
5. Ulangi langkah 1-4 sampai semua kombinasi telah dihitung.

2. Source Program

1. main.cpp

```
1  #include <iostream>
2  #include <cstdlib>
3  #include <ctime>
4  #include <string>
5  #include <vector>
6  #include <limits>
7  #include "stdio.h"
8  #include <stack>
9  #include "string.h"
10 #include <fstream>
11 #include <iterator>
12 #include <chrono>
13 using namespace std;
14 bool isValid(float a, float b, float c, float d);
15 void f2s(float a, float b, float c, float d, std::vector<string> *p);
16 std::string getOP(int x);
17 void getKurbuk(int calc, std::string *kurbuk1, std::string *kurbuk2, std::string *kurbuk3, std::string *kurbuk4, std::string *kurbuk5);
18 void getKurtup(int calc, std::string *kurbuk1, std::string *kurbuk2, std::string *kurbuk3, std::string *kurbuk4, std::string *kurbuk5);
19 void pusVec(int x, int y, int z, int calc, float a, float b, float c, float d, std::vector<std::string> *buffer, bool tf);
20 void swap(float a, float b, float c, float d, float *e, float *f, float *g, float *h, int tag);
21 void removeDupe(std::vector<std::string> *buffer);
22 bool cekStr(std::string token);
23 void getValidInput(std::string a1, std::vector<float> *abcd);
24 std::vector<std::string> split(std::string str, std::string token);
25 void saveFile(std::string name, std::vector<std::string> buffer);
26 void infixToPostfix(string exp, vector<string> *buffer);
27 int getPrecedence(string ops);
28 float getNum(string str);
29 float calculatePostfix(string exp);
```

```

31 ▶ int main(){
32     srand( seed: (int)time( timer: 0));
33     std::vector<std::string> buffer;
34     std::vector<std::string> temp;
35     float a;
36     float b;
37     float c;
38     float d;
39     std::vector<float> abcd;
40     std::string a1="";
41     string menu="";
42     std::cout << "1. input\n2. random\n";
43     std::cin >> menu;
44     while (menu!="1" && menu!="2"){
45         std::cout<<"tidak valid!, ulangi input\n";
46         std::cin >> menu;
47         // std::cin.clear();
48         // std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
49     }
50     if (menu=="1"){
51         getValidInput(a1,&abcd);
52         a=abcd[0];
53         b=abcd[1];
54         c=abcd[2];
55         d=abcd[3];
56     } else if (menu=="2"){
57         a=1+rand()%13;
58         b=1+rand()%13;
59         c=1+rand()%13;
60         d=1+rand()%13;
61         std::cout << "4 angka random:\n";
62         std::cout << a<<" " << b<<" " << c<<" " << d<<"\n";
63     }
64     auto begin :time_point<...> = std::chrono::high_resolution_clock::now();
65     int x=0;

```

```

66     int y=0;
67     int z=0;
68     int calc=0;
69     float atmp,btmp,ctmp,dtmp;
70     atmp=a;
71     btmp=b;
72     ctmp=c;
73     dtmp=d;
74     vector<string> result;
75     int tag=1;
76     // int calc=0;
77     while (tag<25) {
78         x=0;
79         while (x < 4) {
80             y = 0;
81             while (y < 4) {
82                 z = 0;
83                 while (z < 4) {
84                     calc=0;
85                     while (calc<11){
86                         pusVec(x,y,z,calc, a: atmp, b: btmp, c: ctmp, d: dtmp,&buffer, tf: false);
87                         infixtoPostfix( exp: buffer.at( n: buffer.size()-1), buffer: &temp);
88                         if (24- calculatePostfix( exp: temp[temp.size()-1])!=0){
89                             pusVec(x,y,z,calc, a: atmp, b: btmp, c: ctmp, d: dtmp, buffer: &result, tf: true);
90                         }
91                         buffer.pop_back();
92                         temp.pop_back();
93                         calc++;
94                     }
95                     z++;
96                 }
97                 y++;
98             }
99             x++;
100     }

```

```

101 //      printf("%f %f %f %f\n", atmp, btmp, ctmp, dtmp);
102      swap(a,b,c,d, e: &atmp, f: &btmp, g: &ctmp, h: &dtmp, tag);
103 //      printf("%f %f %f %f\n", atmp, btmp, ctmp, dtmp);
104      tag++;
105  }
106  removeDupe( buffer: &result);
107  int i=0;
108  int n=result.size();
109  printf( format: "%d solusi\n",n);
110  while (i<result.size()){
111      std::cout<<result[i]<<"\n";
112      i++;
113  }
114  auto end :time_point<...> = std::chrono::high_resolution_clock::now();
115  auto elapsed :duration<...> = std::chrono::duration_cast<std::chrono::nanoseconds>( d: end - begin);
116
117  string save="";
118  std::cout << "simpan file solusi?\n"<<"1. simpan\n2. tidak\n";
119  std::cin >> save;
120  while (save!="1" && save!="2"){
121      std::cout<<"tidak valid!, ulangi input\n";
122      std::cin >> save;
123      //      std::cin.clear();
124      //      std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
125  }
126  if (save=="1"){
127      std::cout<<"masukkan nama file\n";
128      std::string fileName;
129      std::cin>>fileName;
130      saveFile( name: fileName, buffer: result);
131      //      saveFile(fileName+"1",temp);
132      //      saveFile(fileName+"2",result);
133  }
134  printf( format: "waktu eksekusi: %.3f detik.\n", elapsed.count() * 1e-9);
135  return 0;

```

```

138 void getValidInput(std::string a1, std::vector<float> *abcd){
139     std::string atmp;
140     std::vector<std::string> p;
141     bool isValid= false;
142     std::cout<<"masukkan 4 angka\n";
143     while (!isValid){
144         // cout<<"1";
145         getline( &: std::cin, &: a1);
146         if (a1.empty()){
147             isValid= false;
148             continue;
149         }
150         else{
151             // cout<<"1";
152             atmp=a1;
153             p= split( str: atmp, token: " ");
154             // for (int i = 0; i < p.size(); ++i) {
155             //     cout<<p[i]<<"\n";
156             // }
157             // cout<<p[0]<<"\n"<<p[1]<<"\n"<<p[2]<<"\n"<<p[3]<<"\n";
158             for (int i = 0; i < 4; ++i) {
159                 if (!cekStr( token: p[i])){
160                     isValid= false;
161                     break;
162                 } else{
163                     isValid= true;
164                 }
165             }
166         }
167         if (!isValid){
168             std::cout<<"input tidak valid\n";
169         }
170     }
171     int i=0;
172     while(i<4){

```

```

172     while(i<4){
173         try {
174             abcd->push_back((float ) stoi( str: p[i]));
175         } catch(std::invalid_argument) {
176             if (p[i]=="A"){
177                 abcd->push_back(1);
178             } else if(p[i]=="J"){
179                 abcd->push_back(11);
180             } else if(p[i]=="Q"){
181                 abcd->push_back(12);
182             } else if(p[i]=="K"){
183                 abcd->push_back(13);
184             }
185         }
186         i++;
187     }
188 }
189 // while(((a1->size())>1 || a1->size()<1) || (b1->size())>1 || b1->size()<1) || (c1->size())>1 || c1->size()<1) || (d1->
190 //     cout<<"input tidak valid";
191 //     cin >> *a1 >> *b1 >> *c1 >> *d1;
192 // }
193
194 }

```

```

196 void saveFile(std::string name, std::vector<std::string> buffer){
197     std::string path = "../test/" + name + ".txt";
198     std::ofstream output_file(path);
199     std::ostream_iterator<std::string> output_iterator(&output_file, "\n");
200     std::copy(buffer.begin(), buffer.end(), output_iterator);
201 }
202
203 std::vector<std::string> split(std::string str, std::string token){
204     std::vector<std::string> result;
205     while(str.size()){
206         int index = str.find(token);
207         if(index != std::string::npos){
208             result.push_back(str.substr(0, index));
209             str = str.substr(index + token.size());
210             if(str.size() == 0) result.push_back(str);
211         } else {
212             result.push_back(str);
213             str = "";
214         }
215     }
216     return result;
217 }
218
219
220 bool cekStr(std::string token){
221     return (token=="A")||(token=="2")||(token=="3")||(token=="4")||(token=="5")||(token=="6")||(token=="7")||(token=="8")||(token=="9")||(token=="10")||(token=="J")||(token=="Q")||(token=="K");
222 }
223
224 bool isValid(float a, float b, float c, float d){
225     return (a>0&&a<14)&&(b>0&&b<14)&&(c>0&&c<14)&&(d>0&&d<14);
226 }

```

```

264 void swap(float a, float b, float c, float d, float *e, float *f, float *g, float *h, int tag){
265     if (tag==0){
266         *e=a;
267         *f=b;
268         *g=c;
269         *h=d;
270     } else if (tag==1){
271         *e=a;
272         *f=d;
273         *g=b;
274         *h=c;
275     } else if (tag==2){
276         *e=a;
277         *f=d;
278         *g=b;
279         *h=c;
280     } else if (tag==3){
281         *e=a;
282         *f=b;
283         *g=d;
284         *h=c;
285     } else if (tag==4){
286         *e=a;
287         *f=c;
288         *g=b;
289         *h=d;
290     } else if (tag==5){
291         *e=a;
292         *f=c;
293         *g=d;
294         *h=b;
295     } else if (tag==6){
296         *e=a;
297         *f=d;
298         *g=c;

```



```
295     } else if (tag==6){
296         *e=a;
297         *f=d;
298         *g=c;
299         *h=b;
300     } else if (tag==7){
301         *e=b;
302         *f=a;
303         *g=c;
304         *h=d;
305     } else if (tag==8){
306         *e=b;
307         *f=c;
308         *g=a;
309         *h=d;
310     } else if (tag==9){
311         *e=b;
312         *f=c;
313         *g=d;
314         *h=a;
315     } else if (tag==10){
316         *e=b;
317         *f=d;
318         *g=a;
319         *h=c;
320     } else if (tag==11){
321         *e=b;
322         *f=a;
323         *g=d;
324         *h=c;
325     } else if (tag==12){
326         *e=b;
327         *f=d;
328         *g=c;
329         *h=a;
```

```
330     } else if (tag==13){
331         *e=c;
332         *f=b;
333         *g=d;
334         *h=a;
335     } else if (tag==14){
336         *e=c;
337         *f=d;
338         *g=b;
339         *h=a;
340     } else if (tag==15){
341         *e=c;
342         *f=d;
343         *g=a;
344         *h=b;
345     } else if (tag==16){
346         *e=c;
347         *f=b;
348         *g=a;
349         *h=d;
350     } else if (tag==17){
351         *e=c;
352         *f=a;
353         *g=b;
354         *h=d;
355     } else if (tag==18){
356         *e=c;
357         *f=a;
358         *g=d;
359         *h=b;
360     } else if (tag==19){
361         *e=d;
362         *f=c;
363         *g=a;
364         *h=b;
365     } else if (tag==20){
366         *e=d;
367         *f=a;
368         *g=c;
369         *h=b;
370     } else if (tag==21){
371         *e=d;
372         *f=a;
373         *g=b;
374         *h=c;
375     } else if (tag==22){
376         *e=d;
377         *f=c;
378         *g=b;
379         *h=a;
380     } else if (tag==23){
381         *e=d;
382         *f=b;
383         *g=c;
384         *h=a;
385     } else if (tag==24) {
386         *e = d;
387         *f = b;
388         *g = a;
389         *h = c;
390     }
391 }
```

```

395 void pusVec(int x,int y,int z,int calc,float a, float b, float c, float d, std::vector<std::string> *buffer,bool tf){
396     std::vector<std::string> p;
397     f2s(a,b,c,d,&p);
398     std::string op1,op2,op3;
399     op1=getOP(x);
400     op2=getOP( x y);
401     op3=getOP( x z);
402     std::string kurbuk1,kurbuk2,kurbuk3,kurbuk4,kurbuk5;
403     getKurbuk(calc,&kurbuk1,&kurbuk2,&kurbuk3,&kurbuk4,&kurbuk5);
404     std::string kurtup1,kurtup2,kurtup3,kurtup4,kurtup5;
405     getKurtup(calc, kurbuk1: &kurtup1, kurbuk2: &kurtup2, kurbuk3: &kurtup3, kurbuk4: &kurtup4, kurbuk5: &kurtup5);
406     // string cc= to_string(calc);
407     // string xx= to_string(x);
408     // string yy= to_string(y);
409     // string zz= to_string(z);
410     // cc+ ". "+xx+ " "+yy+ " "+zz+ " "+
411     if (tf){
412         int i=0;
413         while (i<4){
414             if (p[i]=="S"){
415                 p[i]="10";
416             }
417             i++;
418         }
419     }
420     std::string buf=kurbuk1+kurbuk2+p[0]+op1+kurbuk3+kurbuk4+ p[1]+kurtup1+op2+kurbuk5+p[2]+kurtup2+kurtup3+op3+p[3]+kurtup4+kurtup5;
421     std::string s(buf);
422
423
424     // for (int i = 0; i < s.size(); i++) {
425     //     cout << i << ". " << s[i] << "\n";
426     // }
427     buffer->push_back(s);
428 }
429
430 std::string getOP(int x){
431     if (x==0){
432         return "+";
433     } else if (x==1){
434         return "-";
435     } else if (x==2){
436         return "*";
437     } else if (x==3){
438         return "/";
439     } else{
440         return "0";
441     }
442 }

```

```

444 void getKurbuk(int calc, std::string *kurbuk1, std::string *kurbuk2, std::string *kurbuk3, std::string *kurbuk4, std::string *kurbuk5){
445     *kurbuk1="";
446     *kurbuk2="";
447     *kurbuk3="";
448     *kurbuk4="";
449     *kurbuk5="";
450     if (calc==0){ //a b c d
451     } else if (calc==1){ //(a b) c d
452         *kurbuk1="(";
453     } else if (calc==2){ //(a b) (c d)
454         *kurbuk1="(";
455         *kurbuk5="(";
456     } else if (calc==3){ (((a b) c) d
457         *kurbuk1="(";
458         *kurbuk2="(";
459     } else if (calc==4){ // a b (c d)
460         *kurbuk5="(";
461     } else if (calc==5){ // a (b c) d
462         *kurbuk3="(";
463     } else if (calc==6){ // (a (b c)) d
464         *kurbuk1="(";
465         *kurbuk3="(";
466     } else if (calc==7){ // a (b (c d))
467         *kurbuk3="(";
468         *kurbuk5="(";
469     } else if (calc==8){ // a ((b c) d)
470         *kurbuk3="(";
471         *kurbuk4="(";
472     } else if (calc==9){ // a (b c d)
473         *kurbuk3="(";
474     } else if (calc==10){ // (a b c) d
475         *kurbuk1="(";
476     }
477 }

479 void getKurtup(int calc, std::string *kurtup1, std::string *kurtup2, std::string *kurtup3, std::string *kurtup4, std::string *kurtup5){
480     *kurtup1="";
481     *kurtup2="";
482     *kurtup3="";
483     *kurtup4="";
484     *kurtup5="";
485     if (calc==0){ //a b c d
486     } else if (calc==1){ //(a b) c d
487         *kurtup1=")";
488     } else if (calc==2){ //(a b) (c d)
489         *kurtup1=")";
490         *kurtup5=")";
491     } else if (calc==3){ (((a b) c) d
492         *kurtup1=")";
493         *kurtup3=")";
494     } else if (calc==4){ // a b (c d)
495         *kurtup4=")";
496     } else if (calc==5){ // a (b c) d
497         *kurtup3=")";
498     } else if (calc==6){ // (a (b c)) d
499         *kurtup2=")";
500         *kurtup3=")";
501     } else if (calc==7){ // a (b (c d))
502         *kurtup4=")";
503         *kurtup5=")";
504     } else if (calc==8){ // a ((b c) d)
505         *kurtup3=")";
506         *kurtup5=")";
507     } else if (calc==9){ // a (b c d)
508         *kurtup4=")";
509     } else if (calc==10){ // (a b c) d
510         *kurtup3=")";
511     }
512 }

```

```

514 void infixToPostfix(string exp, vector<string> *buffer){
515     vector<string> operatorr;
516     string ret="";
517     string s="";
518     int i=0;
519     while(i<exp.size()){
520         s="";
521         s+=exp.at( i);
522     L1:
523         if (s=="+" || s=="-" || s=="*" || s=="/" || s=="(" || s==")"){
524             if (operatorr.empty()){
525                 operatorr.push_back(s);
526             } else{
527                 if (s=="("){
528                     operatorr.push_back(s);
529                 } else if (s==")"){
530                     while (!operatorr.empty() && operatorr.at( i operatorr.size()-1)!="("){
531                         ret+=operatorr.at( i operatorr.size()-1);
532                         operatorr.pop_back();
533                     }
534                     //
535                     operatorr.pop_back();
536                     if (!operatorr.empty()&&operatorr.at( i operatorr.size()-1)!="("){
537                         operatorr.pop_back();
538                     }
539                 }
540                 else if (getPrecedence( ops operatorr.at( i operatorr.size()-1))< getPrecedence( ops s)){
541                     operatorr.push_back(s);
542                 } else if (getPrecedence( ops operatorr.at( i operatorr.size()-1))== getPrecedence( ops s)){
543                     ret+=operatorr.at( i operatorr.size()-1);
544                     operatorr.pop_back();
545                     operatorr.push_back(s);
546                 } else if (getPrecedence( ops operatorr.at( i operatorr.size()-1))> getPrecedence( ops s)){
547                     // while (!operatorr.empty()&& operatorr.at(operatorr.size()-1)!="(" && getPrecedence(operatorr.at(operatorr.size()-1))> getPrecedence(s)){
548                     //     ret+=operatorr.at(operatorr.size()-1);
549                     //     operatorr.pop_back();
550                     // }
551                     // if (!operatorr.empty()&&operatorr.at(operatorr.size()-1)!="("){
552                     //     operatorr.pop_back();
553                     // }
554                     ret+=operatorr.at( i operatorr.size()-1);
555                     operatorr.pop_back();
556                     goto L1;
557                 }
558             }
559         } else {
560             if (s!="("&&s!=")")){
561                 ret+=s;
562             }
563         }
564         i++;
565     }
566     while (!operatorr.empty()){
567         ret+=operatorr.at( i operatorr.size()-1);
568         operatorr.pop_back();
569     }
570     buffer->push_back(ret);
571 }

```

```

573 int getPrecedence(string ops){
574     if (ops=="+" || ops=="-"){
575         return 1;
576     } else if (ops=="*" || ops=="/"){
577         return 2;
578     } else {
579         return 0;
580     }
581 }
582
583 float getNum(string str){
584     float f=0;
585     try {
586         f= stof(str);
587     } catch(std::invalid_argument) {
588         if (str=="S"){
589             f=10;
590         } else if (str=="J"){
591             f=11;
592         } else if (str=="Q"){
593             f=12;
594         } else if (str=="K"){
595             f=13;
596         }
597     }
598     return f;
599 }
601 float calculatePostfix(string exp){
602     vector<float> res;
603     string s="";
604     int i=0;
605     float o1=0;
606     float o2=0;
607     float result=0;
608     while (i<exp.size()){
609         s+=exp[i];
610         if (s=="+" || s=="-" || s=="*" || s=="/"){
611             o2=res[res.size()-1];
612             res.pop_back();
613             o1=res[res.size()-1];
614             res.pop_back();
615             if (s==""){
616                 result=o1+o2;
617                 res.push_back(result);
618             } else if (s=="-"){
619                 result=o1-o2;
620                 res.push_back(result);
621             } else if (s=="*"){
622                 result=o1*o2;
623                 res.push_back(result);
624             } else if (s=="/"){
625                 result=o1/o2;
626                 res.push_back(result);
627             }
628         } else{
629             res.push_back(getNum( str: s));
630         }
631         i++;
632     }
633     // cout<<res.size()<<"\n";
634     return res[0];
635 }

```

3. Input/Output

1. masukan berdasarkan input (1)

```
1. input
2. random
1
masukkan 4 angka
9 8 A K
12 solusi
8*(K-(9+1))
8*((K-9)-1)
8*(K-9-1)
8*(K-(1+9))
8*((K-1)-9)
8*(K-1-9)
(K-(1+9))*8
((K-1)-9)*8
(K-1-9)*8
(K-(9+1))*8
((K-9)-1)*8
(K-9-1)*8
simpan file solusi?
1. simpan
2. tidak
1
masukkan nama file
testinput1
waktu eksekusi: 0.212 detik.
```

2. masukan berdasarkan input (2)

```
1. input
2. random
1
masukkan 4 angka
2 5 4 5
8 solusi
2*(5+5)+4
(2*(5+5))+4
(5+5)*2+4
((5+5)*2)+4
4+(5+5)*2
4+((5+5)*2)
4+2*(5+5)
4+(2*(5+5))
simpan file solusi?
1. simpan
2. tidak
1
masukkan nama file
testinput2
waktu eksekusi: 0.104 detik.
```

3. masukan berdasarkan input (3)

```
1. input
2. random
1
masukkan 4 angka
6 9 5 K
4 solusi
(6-9)*(5-K)
(9-6)*(K-5)
(5-K)*(6-9)
(K-5)*(9-6)
simpan file solusi?
1. simpan
2. tidak
1
masukkan nama file
testinput3
waktu eksekusi: 0.198 detik.
```

4. masukan random (1)


```

1. input
2. random
2
4 angka random:
5 2 4 12
38 solusi
(5-2)*4+Q
((5-2)*4)+Q
(5-2)*(Q-4)
(5-4)*2*Q
(5-4)*(2*Q)
((5-4)*2)*Q
(5-4)*Q*2
(5-4)*(Q*2)
((5-4)*Q)*2
(2-5)*(4-Q)
2*(5-4)*Q
(2*(5-4))*Q
2*((5-4)*Q)
2/(5-4)*Q
(2/(5-4))*Q
2/((5-4)/Q)
(2*Q)*(5-4)
2*Q*(5-4)
2*(Q*(5-4))
(2*Q)/(5-4)
2*Q/(5-4)
2*(Q/(5-4))
(4-Q)*(2-5)
4*(5-2)+Q
(4*(5-2))+Q
Q+4*(5-2)
Q+(4*(5-2))
(Q-4)*(5-2)
Q*(5-4)*2
(Q*(5-4))*2
Q*((5-4)*2)
Q/(5-4)*2
(Q/(5-4))*2
Q/((5-4)/2)
Q+(5-2)*4
Q+((5-2)*4)
Q-4*(2-5)
Q-(4*(2-5))
simpan file solusi?
1. simpan
2. tidak
1
masukkan nama file
testrandom1
waktu eksekusi: 0.184 detik.

```

5. masukan random (2)

```

1. input
2. random
2
4 angka random:
10 7 1 5
26 solusi
10*((7/5)+1)
10*(7/5+1)
10*(1+(7/5))
10*(1+7/5)
(7*5)-(10+1)
7*5-(10+1)
7*5-10-1
(7*5)-10-1
((7*5)-10)-1
(7*5-10)-1
(7*5)-(1+10)
7*5-(1+10)
7*5-1-10
(7*5)-1-10
((7*5)-1)-10
(7*5-1)-10
((7/5)+1)*10
(7/5+1)*10
(1+(7/5))*10
(1+7/5)*10
(5*7)-(1+10)
5*7-(1+10)
5*7-1-10
(5*7)-1-10
((5*7)-1)-10
(5*7-1)-10
simpan file solusi?
1. simpan
2. tidak
1
masukkan nama file
testrandom2
waktu eksekusi: 0.221 detik.

```

6. masukan random (3)

```

1. input
2. random
2
4 angka random:
5 9 12 12
32 solusi
((5+9)-Q)*Q
(5+(9-Q))*Q
(5+9-Q)*Q
((5-Q)+9)*Q
(5-Q+9)*Q
(5-(Q-9))*Q
((9+5)-Q)*Q
(9+(5-Q))*Q
(9+5-Q)*Q
((9-Q)+5)*Q
(9-Q+5)*Q
(9-(Q-5))*Q
((9*Q)+Q)/5
(9*Q+Q)/5
(Q+(9*Q))/5
(Q+9*Q)/5
((Q*9)+Q)/5
(Q*9+Q)/5
Q*((9-Q)+5)
Q*(9-Q+5)
Q*(9-(Q-5))
(Q+(Q*9))/5
(Q+Q*9)/5
Q*(9+(5-Q))
Q*((9+5)-Q)
Q*(9+5-Q)
Q*(5+(9-Q))
Q*((5+9)-Q)
Q*(5+9-Q)
Q*((5-Q)+9)
Q*(5-Q+9)
Q*(5-(Q-9))
simpan file solusi?
1. simpan
2. tidak
1
masukkan nama file
testrandom3
waktu eksekusi: 0.255 detik.

```

Poin	Ya	Tidak
1. Program berhasil dikompilasi tanpa kesalahan	✓	
2. Program berhasil <i>running</i>	✓	
3. Program dapat membaca input / generate sendiri dan memberikan luaran	✓	
4. Solusi yang diberikan program memenuhi (berhasil mencapai 24)	✓	
5. Program dapat menyimpan solusi dalam file teks	✓	