**LAPORAN TUGAS KECIL I**

**IF2211 STRATEGI ALGORITMA**

**Penyelesaian** Permainan Kartu 24 **dengan Algoritma Brute Force**

****

Oleh

Vieri Fajar Firdaus

13521099

K1

**PROGRAM STUDI TEKNIK INFORMATIKA  
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA  
INSTITUT TEKNOLOGI BANDUNG  
2022**

**BAB 1**

**ALGORITMA *BRUTE FORCE***

Algoritma *brute force* adalah suatu strategi penyelesaian permasalahan komputasi dengan meninjau semua kasus yang mungkin dari permasalahan tersebut. Pencarian solusi permasalahan menggunakan algoritma *brute force* dengan tepat menjamin solusi permasalahan tersebut ditemukan untuk setiap kemungkinan. Namun, penggunaan algoritma ini cukup mahal. Permasalahan dengan kasus yang perlu ditinjau cukup banyak akan memerlukan memori yang besar dan waktu yang lama. Oleh sebab itu, algoritma ini cocok dipakai untuk batasan banyaknya kasus yang perlu ditinjau tidak terlalu banyak.

Ada banyak sekali cara untuk mengoptimalisasi algoritma *brute force*. Sudut pandang  
pengambilan kasus yang akan ditinjau saja tidak jarang mengoptimalisasi program secara signifikan. Umumnya, optimalisasi algoritma *brute force* dilakukan dengan mengurangi kemungkinan kasus yang sudah dipastikan tidak akan menjadi solusi persoalan, sehingga kasus yang perlu ditinjau menjadi lebih sedikit.

Pada penyelesaian permainan kartu 24 ini, langkah pertama yang dilakukan adalah menerima masukan dari pengguna dengan 4 buah kartu dengan ketentuan kartu (A, 2-10, J, Q, K, 11, 12, 13), pada program yang dibuat akan dilakukan validasi input, apabila salah satu input tidak sesuai dengan ketentuan maka akan diulang sampai seluruh input memenuhi ketentuan.

Pada input masukan, akan diberikan pilihan input manual atau input random, pada saat input manual, user akan menginput 4 masukan dan akan divalidasi, sedangkan untuk input random akan digunakan “#include <cstdlib>” untuk melakukan random, setelah itu akan dilakukan permutasi untuk menentukan banyaknya susunan kartu, dalam hal ini digunakan algoritma brute force untuk melakukannya, normalnya akan terdapat 4! Atau 24 kemungkinan tetapi terdapat beberapa kasus sehingga permutasinya akan kurang dari 24.

Setelah diperoleh permutasi urutan kartu, akan dilakukan permutasi untuk operasi (+, -, x, /) dan akan terdapat atau 64 kemungkinan, selain itu terdapat juga susunan untuk peletakan kurung atau grouping dengan 5 kemungkinan dengan susunan:

1. (
2. (

# dengan sebagai angka kartu, dan sebagai operasi.

Setelah dilakukan pencarian semua kemungkinan dengan memperhitungkan permutasi susunan kartu, susunan operasi dan grouping akan diperoleh kemungkinan, untuk setiap solusi akan disimpan dalam sebuah *array of string* dan akan ditampilkan dalam *comman prompt,* dan dapat juga disimpan dalam *file* *txt.*

Untuk waktu perhitungan eksekusi akan dihitung dari awal pencarian susunan permutasi kartu, susunan operasi, grouping, pencarian solusi kartu 24, dan penyimpanan ke dalam txt, waktu akan ditampilkan dalam satuan milisekon. File akan menampilkan banyak solusi dari kartu 24 dan akan menampilkan seluruh kemungkinan untuk mendapat kartu 24

**BAB 2**

***SOURCE PROGRAM***

|  |
| --- |
| #include <bits/stdc++.h>  #include <string.h>  #include <chrono>  #include <cstdlib>  using namespace std;  string input[5];  int a[5],b[5],op[5];  bool cek[5];  int res;  char cop[4]={'+','-','\*','/'};  string cob[]= {"-","A","2","3","4","5","6","7","8","9","10","J","Q","K"};  string convint[]={"-","1","2","3","4","5","6","7","8","9","10","11","12","13"};  bool adasolusi=false;  int permutation[25][5];  int cnt\_permutation;  string solution[10000];  int cnt\_solution;  string tempsave;  char ss;  void conv(){      for(int i=0;i<4;i++){          if(input[i]>="1" && input[i]<="9" && input[i].length()==1){              a[i]=(input[i][0]-'0');          }          else{              if(input[i]=="10"){                  a[i]=10;              }              else if(input[i]=="J" || input[i]=="11"){                  a[i]=11;              }              else if(input[i]=="Q" || input[i]=="12"){                  a[i]=12;              }              else if(input[i]=="K" || input[i]=="13"){                  a[i]=13;              }              else if(input[i]=="A"){                  a[i]=1;              }          }      }  }  double operation(double a,double b,int op){      double temp;      if(op==0){          temp=a+b;      }      else if(op==1){          temp=a-b;      }      else if(op==2){          temp=a\*b;      }      else if(op==3){          temp=double(double(a)/double(b));      }      return temp;  }  double f(int i,int j){      double tempa,tempb,result;  *// (a op b) op (c op d)*      if(i==0){          tempa=operation(permutation[j][0],permutation[j][1],op[0]);          tempb=operation(permutation[j][2],permutation[j][3],op[2]);          result=operation(tempa,tempb,op[1]);      }  *// (a op b) op c) op d*      else if(i==1){          tempa=operation(permutation[j][0],permutation[j][1],op[0]);          tempb=operation(tempa,permutation[j][2],op[1]);          result=operation(tempb,permutation[j][3],op[2]);      }  *// (a op (b op c)) op d*      else if(i==2){          tempa=operation(permutation[j][1],permutation[j][2],op[1]);          tempb=operation(permutation[j][0],tempa,op[0]);          result=operation(tempb,permutation[j][3],op[2]);      }  *// a op ((b op c) op d)*      else if(i==3){          tempa=operation(permutation[j][1],permutation[j][2],op[1]);          tempb=operation(tempa,permutation[j][3],op[2]);          result=operation(permutation[j][0],tempb,op[0]);      }  *// a op (b op (c op d))*      else if(i==4){          tempa=operation(permutation[j][2],permutation[j][3],op[2]);          tempb=operation(permutation[j][1],tempa,op[1]);          result=operation(permutation[j][0],tempb,op[0]);      }      return result;  }  void print\_solutin(int a,int idx){      if(a==0){          solution[cnt\_solution]="(" + convint[permutation[idx][0]] + " " + cop[op[0]] + " " + convint[permutation[idx][1]] + ") " + cop[op[1]] + " (" + convint[permutation[idx][2]] + " " + cop[op[2]] + " " + convint[permutation[idx][3]]+ ")";      }      else if(a==1){          solution[cnt\_solution]="((" + convint[permutation[idx][0]] + " " + cop[op[0]] + " " + convint[permutation[idx][1]] + ") " + cop[op[1]] + " " + convint[permutation[idx][2]] + ") " + cop[op[2]] + " " + convint[permutation[idx][3]];      }      else if(a==2){          solution[cnt\_solution]="(" + convint[permutation[idx][0]] + " " + cop[op[0]] + " (" + convint[permutation[idx][1]] + " " + cop[op[1]] + " " + convint[permutation[idx][2]] + ")) " + cop[op[2]] + " " + convint[permutation[idx][3]];      }      else if(a==3){          solution[cnt\_solution]=convint[permutation[idx][0]] + " " + cop[op[0]] + " ((" + convint[permutation[idx][1]] + " " + cop[op[1]] + " " + convint[permutation[idx][2]]+ ") " + cop[op[2]] + " " + convint[permutation[idx][3]] + ")";      }      else if(a==4){          solution[cnt\_solution]=convint[permutation[idx][0]] + " " + cop[op[0]] + " (" + convint[permutation[idx][1]] + " " + cop[op[1]] + " (" + convint[permutation[idx][2]]+ " " + cop[op[2]] + " " + convint[permutation[idx][3]]+ "))";      }      cnt\_solution++;  }  bool cek\_in(){      bool cek=true;      for(int i=0;i<cnt\_permutation;i++){          if(permutation[i][0]==b[0] && permutation[i][1]==b[1] && permutation[i][2]==b[2] && permutation[i][3]==b[3]){              return false;              cek=false;              break;          }      }      return cek;  }  void per\_number(int first,int last){      if(first==last){          if(cek\_in()){              permutation[cnt\_permutation][0]=b[0];              permutation[cnt\_permutation][1]=b[1];              permutation[cnt\_permutation][2]=b[2];              permutation[cnt\_permutation][3]=b[3];              cnt\_permutation++;          }      }      else {          for(int i=0;i<4;i++){              if(cek[i]==false){                  b[first]=a[i];                  cek[i]=true;                  per\_number(first+1,last);                  cek[i]=false;              }          }      }  }  void cek\_24(){      for(int j=0;j<cnt\_permutation;j++){          for(int i=0;i<5;i++){              if(f(i,j)-24>-0.00000000001 && f(i,j)-24<0.00000000001){  *// cout<<f(i,j)<<endl;*                  if(adasolusi==false){                      cout<<"Berikut adalah solusi dari permainan kartu 24"<<endl;                  }                  adasolusi=true;                  res++;                  print\_solutin(i,j);              }          }      }    }  void per\_op(int first,int last){      if(first==last){          cek\_24();      }      else{          for(int i=0;i<4;i++){              op[first]=i;              per\_op(first+1,last);          }      }  }  bool cekinp(string s){      if(s=="2"||s=="3"||s=="4"||s=="5"||s=="6"||s=="7"||s=="8"||s=="9"||s=="10"||s=="J"||s=="Q"||s=="K"||s=="A" || s=="11"||s=="12"||s=="13"||s=="1"){          return true;      }      else{          return false;      }  }  int main(){  *// mempercepat input output*      ios\_base::sync\_with\_stdio(false);      cin.tie(NULL);  *// input angka*      bool cekinput;      cekinput=true;      int random;      cout<<"Masukkan input random atau manual? "<<endl;      cout<<"1 untuk random"<<endl<<"2 untuk manual"<<endl;      cin>>random;        while(random!=1 && random!=2){          cout<<"Input salah, ulangi"<<endl;          cin>>random;      }      if(random==1){          srand(time(0));          for(int i=0;i<4;i++){              input[i]=cob[rand()%13+1];          }          cout<<"input berupa"<<endl;          cout<<input[0]<<" "<<input[1]<<" "<<input[2]<<" "<<input[3]<<endl;      }      else if(random==2){          while(cekinput){              cout<<"Masukan input berupa 4 angka"<<endl;              cin>>input[0]>>input[1]>>input[2]>>input[3];              for(int i=0;i<4;i++){                  if(cekinp(input[i])){                      cekinput=false;                  }                  else{                      cout<<"Input salah, ulangi"<<endl;                      cekinput=true;                      break;                  }              }          }      }    *// kode utama*      auto awal = std::chrono::high\_resolution\_clock::now();      conv();      per\_number(0,4);      per\_op(0,3);      if(adasolusi==false){          cout<<"Tidak ada solusi"<<endl;      }      else{          cout<<"Jumlah solusi : "<<res<<endl;      }      for(int i=0;i<cnt\_solution;i++){          cout<<solution[i]<<endl;      }      auto akhir = std::chrono::high\_resolution\_clock::now();        cout<<endl<<"Apakah anda ingin menyimpan hasil ke file? (y/n)"<<endl;      cin>>ss;        while(ss!='y' && ss!='n'){          cout<<"Input salah, ulangi"<<endl;          cin>>ss;      }      if(ss=='y'){          cout<<endl<<"Masukkan nama file : "<<endl;          cin>>tempsave;      }      auto awalsave = std::chrono::high\_resolution\_clock::now();      if(ss=='y'){          if(cnt\_solution==0){              ofstream MyFile("../test/"+tempsave+".txt");              MyFile << "Tidak ada solusi"<<endl;              MyFile.close();          }          else{              ofstream MyFile("../test/"+tempsave+".txt");              MyFile<<"Berikut adalah solusi dari permainan kartu 24 dengan input"<<endl;              MyFile<<input[0]<<" "<<input[1]<<" "<<input[2]<<" "<<input[3]<<endl<<endl;              MyFile<<"Jumlah solusi : "<<cnt\_solution<<endl;              for(int i=0;i<cnt\_solution;i++){                  MyFile << solution[i]<<endl;              }              MyFile.close();          }      }      auto akhirsave = std::chrono::high\_resolution\_clock::now();      auto elapsed = std::chrono::duration\_cast<std::chrono::nanoseconds>(akhir - awal+akhirsave-awalsave);      cout<<"Waktu eksekusi selama: "<<elapsed.count() \* 1e-6<<" milidetik."<<endl;        return 0;  }  *//g++ code.cpp -o code* |

**BAB 3**

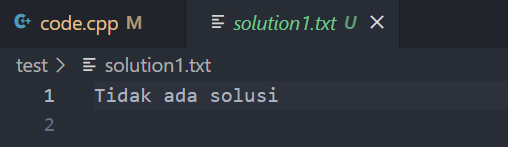
**TANGKAPAN LAYAR PENGETESAN**

1. Susunan dengan input manual dengan solusi tidak ada.
2. Contoh 1

Text

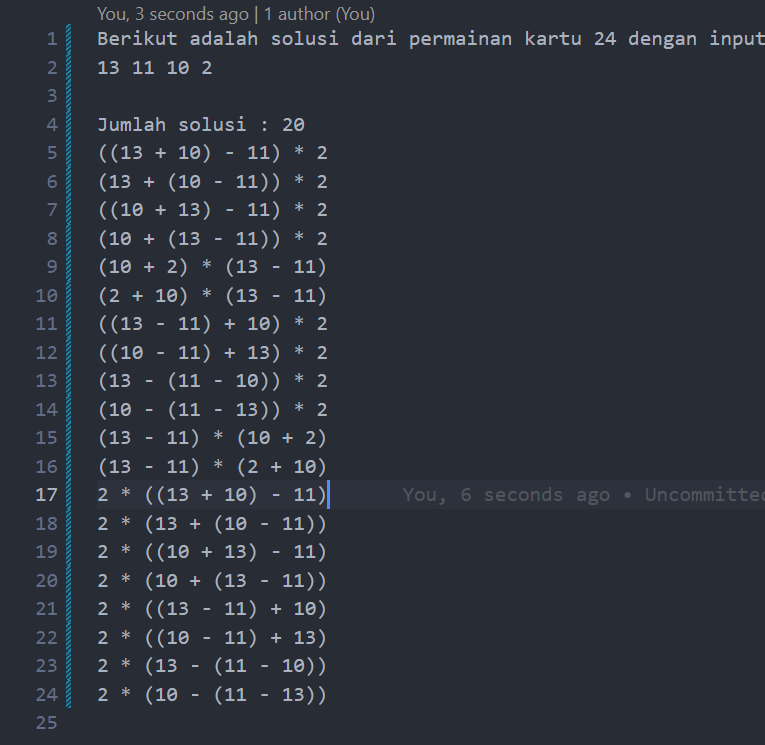
Description automatically generated

Berikut untuk tampilan txt hasil penyimpanan

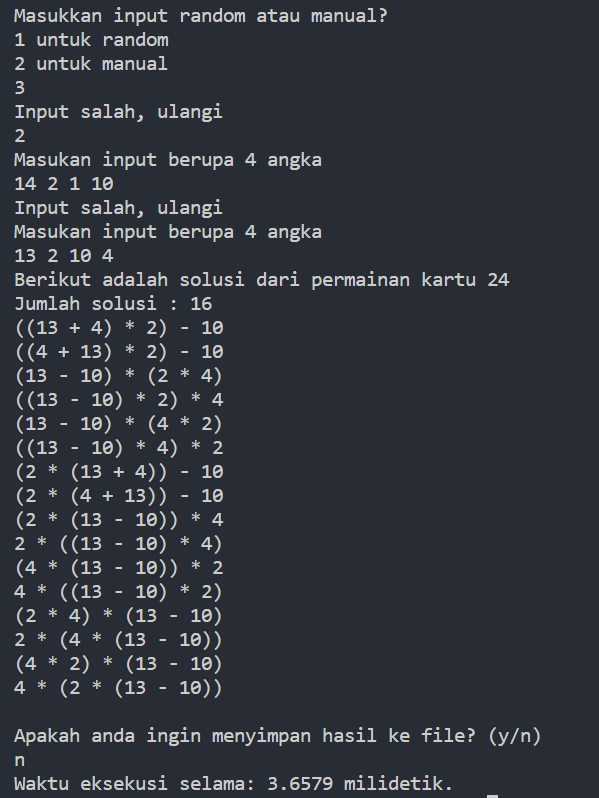


1. Susunan dengan input manual dengna solusi ada.
2. Text

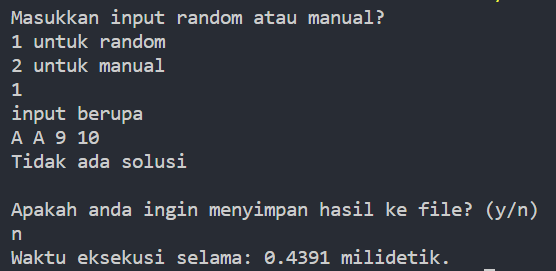
   Description automatically generatedContoh 1



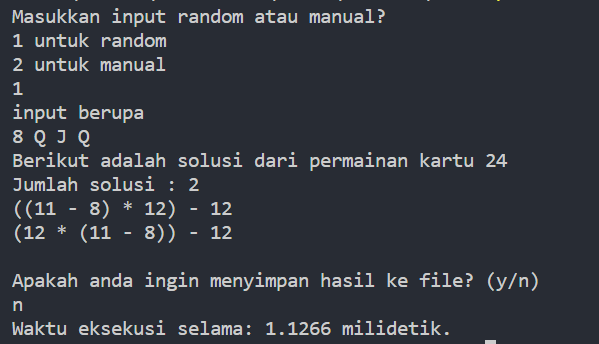
1. Contoh 2



1. Susunan dengan input random.
2. Contoh 1



1. Contoh 2



**BAB 4**

**DOKUMENTASI DAN *CHECKLIST***

1. Repository GitHub

<https://github.com/vierifirdaus/Tucil1_13521099>

1. *Checklist*

|  |  |  |
| --- | --- | --- |
| Poin | Ya | Tidak |
| Program berhasil dikompilasi tanpa kesalahan | Check mark, Wingdings font, character code 252 decimal. |  |
| Program berhasil *running* | Check mark, Wingdings font, character code 252 decimal. |  |
| Program dapat membaca input / generate sendiri dan memberikan luaran | Check mark, Wingdings font, character code 252 decimal. |  |
| Solusi yang diberikan program memenuhi (berhasil mencapai 24) | Check mark, Wingdings font, character code 252 decimal. |  |
| Program dapat menyimpan solusi dalam file teks | Check mark, Wingdings font, character code 252 decimal. |  |