# Laporan Tugas

## Program DFA Marble Rolling Toy

### IF2220 Teori Bahasa Formal dan Otomata

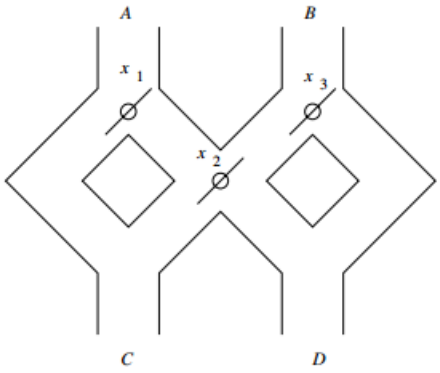
**Disusun oleh**
**Fadhil Imam Kurnia – 13515146**

**Program Studi Teknik Informatika**

**Sekolah Teknik Elektro dan Informatika**

**Institut Teknologi Bandung**

## A. Deskripsi Persoalan

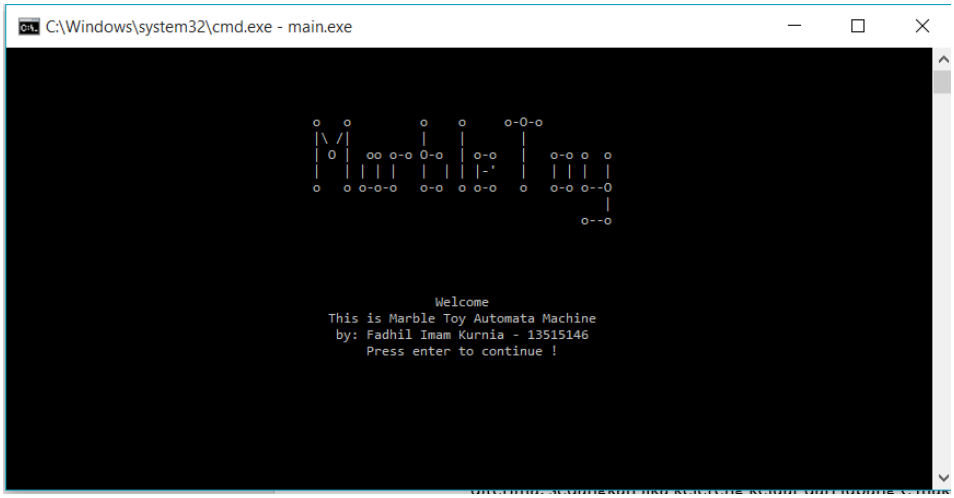



*Gambar 1 Ilustrasi Marble Rolling Toy*

Marble Rolling Toy merupakan sebuah perangkat permainan yang memiliki banyak kemungkinan. Bentuk Marble Rolling Toy sendiri dapat dilihat pada gambar 1. Cara memainkannya adalah dengan menjatuhkan kelereng melalui lubang A ataupun lubang B. Setelah dijatuhkan dari lubang A atau B kemudian kelereng akan meluncur turun menelusuri Marble Rolling Toy. Didalam Marble Rolling Toy terdapat 3 buah tuas yang memiliki orientasi kanan dan kiri. Setiap kali kelereng menyentuh tuas, orientasi tuas akan berganti. Semisal tuas x1 pada awalnya berorientasi ke kiri, setelah dijatuhkan kelereng dari lubang A dan kelereng tersebut mengenai tuas x1 maka orientasi tuas x1 akan berganti menjadi ke kanan.

Terdapat 2 tempat keluar untuk kelereng dalam Marble Rolling Toy. Tempat keluar tersebut adalah lubang C dan lubang D. Jika kelereng keluar dari lubang D maka statusnya adalah diterima, sedangkan jika kelereng keluar dari lubang C maka statusnya adalah ditolak.

Terkadang cukup sulit bagi kita untuk menentukan apakah *state* yang terjadi akan diterima atau ditolak. Kita perlu membayangkan alur berjalannya kelereng dalam permainan tersebut. Oleh karena itu akan dibuat sebuah program yang mensimulasikan gerak kelereng dalam permainan tersebut. Program yang dibuat juga akan menentukan apakah status yang terjadi saat di akhir permainan tersebut ditolak atau diterima.




*Gambar 2 Antarmuka program Marble Toy*

Program yang dibuat oleh penulis diberi nama “Marble Toy” dan dikembangkan menggunakan bahasa pemrograman prosedural C. Tampilan program dapat dilihat pada gambar 2. Penulis juga mendesain antar muka program yang dibuat agar terlihat menarik dan mudah digunakan. Untuk dapat menggunakan program tersebut kita cukup mengikuti instruksi yang ada.

Tentu saja program yang dibuat oleh penulis masih memiliki banyak kekurangan. Namun penulis akan berusaha untuk membuat program tersebut menjadi lebih baik. Oleh karena itu saran dan masukan yang membangun untuk pengembangan program tersebut sangat berarti bagi penulis. *Source code* program akan penulis lampirkan dalam Bab C. Selain itu *sorce code* program juga dapat dilihat dan diunduh melalui link :

https://github.com/fadhilimamk/Marbel-Toy-Machine

Saran dan masukan dapat saya terima melalui *issues* pada link tersebut juga.

## B. DFA

Jika kita perhatikan dengan baik, kita dapat memodelkan permainan Marble Alfabet yang diterima oleh DFA yang merepresentasikan permainan tersebut adalah A, dan B. Alfabet tersebut menggambarkan tempat masuknya kelereng kedalam Marble Rolling Toy. Terdapat banyak *state* yang mungkin dalam DFA permainan tersebut, jika dihutung terdapat $2^4$ atau 16 kemungkinan yang dapat terjadi, namun hanya ada 13 *state* yang terjadi dalam permainan tersebut.

Dalam DFA yang dibuat kita dapat merepresentasikan 0 sebagai tanda bahwa tuas sedang mengarah ke kiri, dan 1 merepresentasikan tanda bahwa tuas sedang mengarah ke kanan. Setiap *state* dapat direpresentasikan sebagai urutan 3 karakter 0 atau 1, diikuti oleh karakter a atau r. Karakter a berarti status akhir permainan adalah diterima (*accepted*) sedangkan r menandakan status akhir permainan ditolak (*rejected*). Sebagai contoh 001r menunjukan bahwa tuas x1 sedang mengarah ke kiri, tuas x2 sedang mengarah ke kiri, tuas x3 sedang mengarah ke kanan, dan status saat itu adalah ditolak. *State* awal dalam permainan ini adalah 000r. Berikut tabel transisi *state* yang mungkn terjadi :

| | A | B |
|---|---|---|
| ->000r | 100r | 011r |
| *000a | 100r | 011r |
| *001a | 101r | 000a |
| 010r | 110r | 001a |
| *010a | 110r | 001a |
| 011r | 111r | 010a |
| 100r | 010r | 111r |
| *100a | 010r | 111r |
| 101r | 011r | 100a |
| *101a | 011r | 100a |
| 110r | 000a | 101a |
| *110a | 000a | 101a |
| 111r | 001a | 110a |

Notasi lengkap DFA permainan ini adalah:

∑ = {A,B}

Q = {000r,000a,001a,010r,010a,011r,100r,100a,101r,101a,110r,110a,111r}

$S / Q_0$ = 000r

δ

F = {000a,001a,010a,100a,101a,110a}

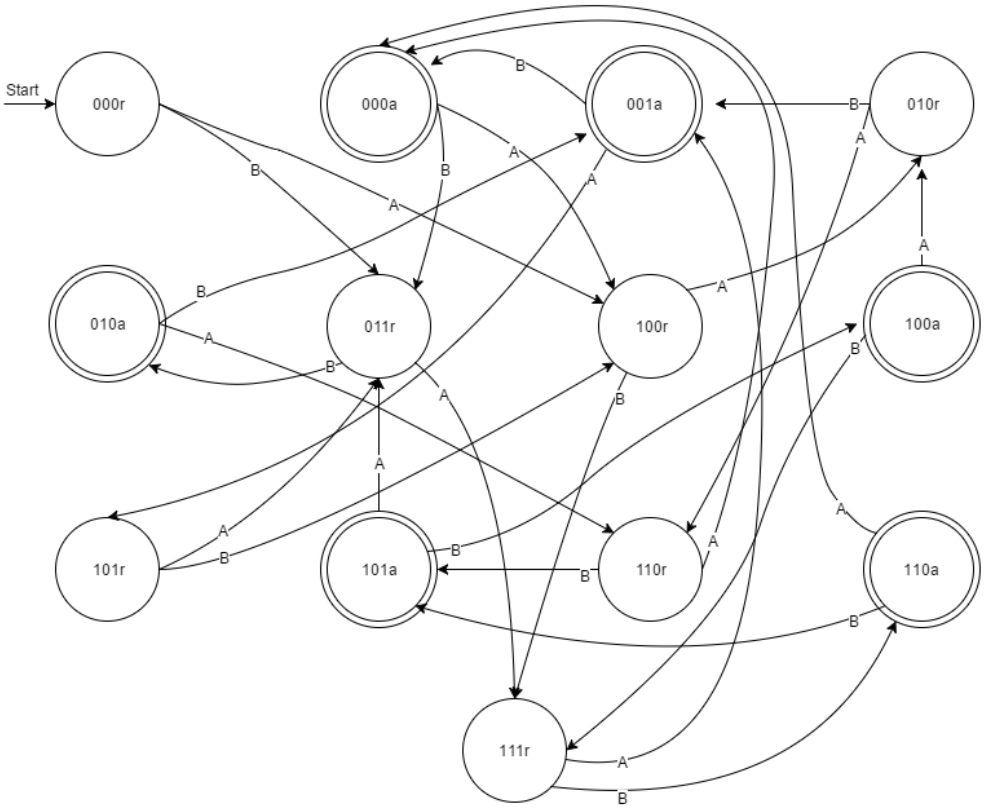
Kita dapat menyatakan notasi tersebut dengan himpunan berikut:

A = {

{000r,000a,001a,010r,010a,011r,100r,100a,101r,101a,110r,110a,111r},
{A,B},
δ,
000r,
{000a,001a,010a,100a,101a,110a}

}

Dari tabel transisi dapat dibuat diagram DFA seperti berikut:



**C. *Source Code***

Seperti yang sudah diterangkan sebelumnya, *source code* program ini dapat diunduh melalui link: https://github.com/fadhilimamk/Marbel-Toy-Machine. Struktur folder dan file pada proyek program ini adalah sebagai berikut:

```
---- bin
--------- main.exe
--------- readme.txt
---- doc
--------- laporan.pdf
---- src
--------- data
---------- language.dfa
--------- controller.c
--------- main.c
--------- model.c
--------- view.c
--------- boolean.h
--------- data
--------- controller.h
--------- model.h
--------- view.h
```

Berikut *source code* program yang penulis telah buat :

File : main.c

```
/*
Marble Toy Automata Machine
Author  : Fadhil Imam Kurnia - 13515146
Date    : 13 September 2016
File    : main.c v1.00
Desc    : Machine to accept string and show whether that string accepted or not
        base on state that saved on external file
*/

#include <stdio.h>
#include "view.h"
#include "model.h"
#include <conio.h>
#include "boolean.h"
#include "controller.h"

int main(){
  DFA dfa;
  char kalimat[256];

  ShowStartPage();
  clearScreen();
  ShowHeader();
  printf(" # Kondisi Awal Marble Toy : \n");
```

```
  ShowToy(true,true,true);
  ReadFile(&dfa);
  printf(" # Masukkan kalimat: ");
  scanf("%s", &kalimat);
  printf("\n # Proses: ");
  proses(&dfa, kalimat);
  printf(" # Hasil akhir: >>> %s ", prosesFinal(&dfa, kalimat));

  if (isAccepted(prosesFinal(&dfa, kalimat))) {
    printf("(DITERIMA)");
  }else{
    printf("(DITOLAK)");
  }

  printf("\n # Kondisi Akhir Marble Toy : \n");
  ShowToy(prosesFinal(&dfa, kalimat)[0],prosesFinal(&dfa, kalimat)[1],prosesFinal(&dfa,
kalimat)[2]);


  getch();

  return 0;
}
```

File : model.c

```
/*
Marble Toy Automata Machine
Author  : Fadhil Imam Kurnia - 13515146
Date    : 18 September 2016
File    : model.c v1.00
Desc    : File to support main app, connecting with data
*/


#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "model.h"

void ReadFile(DFA * dfa){
  FILE *file;
  char buff[255];
  char temp[255];
  char alphabet[255];
  int i,j=0,count = 0;


file = fopen("..//src//data//language.dfa", "r");
  // baca alfabet
  fgets(buff, 255, (FILE*)file);
  strncpy(temp, buff, 12  );temp[12]='\0';
```

```
if (strcmp(temp,"alphabet : {")==0) {
  i = 12;
  while(buff[i]!='}'){
    if (buff[i]!=',') {
      (*dfa).Alphabet[count]=buff[i];
      count++;
    }
    i++;
  }
  (*dfa).countAlphabet = count;
}else{
  printf("terjadi kesalahan saat pembacaan file\n");
  exit(0);
}

// baca state
fgets(buff, 255, (FILE*)file);count=0;
strncpy(temp, buff, 9  );temp[9]='\0';
if (strcmp(temp,"state : {")==0) {
  memset(temp, 0, sizeof(temp));  //reset memori temp karena akan dipakai ulang
  i=9;
  while (buff[i]!='}') {
    if(buff[i]!=','){
      temp[j] = buff[i];
      j++;
    }else{
      j=0;
      (*dfa).State[count] = (char*)malloc(sizeof(temp));
      strcpy((*dfa).State[count], temp);
      count++;
    }
    i++;
  }
  (*dfa).countState = count +1;
}else{
  printf("terjadi kesalahan saat pembacaan file\n");
  exit(0);
}

//baca Start
fgets(buff, 255, (FILE*)file);count=0;
strncpy(temp, buff, 8  );temp[8]='\0';
if (strcmp(temp,"start : ")==0) {
  memset(temp, 0, sizeof(temp));
  memcpy( (*dfa).Start, &buff[8], strlen(buff) );
}else{
  printf("terjadi kesalahan saat pembacaan file\n");
  exit(0);
}

// baca final
```

```
fgets(buff, 255, (FILE*)file);count=0;
strncpy(temp, buff, 9  );temp[9]='\0';
if (strcmp(temp,"final : {")==0) {
  memset(temp, 0, sizeof(temp));  //reset memori temp karena akan dipakai ulang
  i=9;j=0;
  while (buff[i]!='}') {
    if(buff[i]!=','){
      temp[j] = buff[i];
      j++;
    }else{
      j=0;
      (*dfa).Final[count] = (char*)malloc(sizeof(temp));
      strcpy((*dfa).Final[count], temp);
      count++;
    }
    i++;
  }
  (*dfa).countDelta = count;
}else{
  printf("terjadi kesalahan saat pembacaan file\n");
  exit(0);
}

// baca delta
fgets(buff, 255, (FILE*)file);count=0;
strncpy(temp, buff, 7  );temp[7]='\0';
if (strcmp(temp,"delta :")==0) {
  boolean a = false, b= false,c = false;
  fgets(buff, 255, (FILE*)file);
  count=0;
  while(!feof((FILE*)file)){
    memset(temp, 0, sizeof(temp));
    i=1;j=0;
    while(buff[i]!=')'){
      if(buff[i]=='|'){
        strcpy((*dfa).delta[count].istate, temp);
        j=0;
      }
      if(buff[i]=='='){
        (*dfa).delta[count].sigma = temp[j-1];
        j=0;
        i++;
      }
      temp[j] = buff[i];
      i++;j++;
    }
    strcpy((*dfa).delta[count].fstate, temp);
    count++;
    fgets(buff, 255, (FILE*)file);
  }
  (*dfa).countDelta = count;
```

```
  }else{
    printf("terjadi kesalahan saat pembacaan file\n");
    exit(0);
  }


  fclose(file);

}
```

File : controller.c

```
/*
Marble Toy Automata Machine
Author  : Fadhil Imam Kurnia - 13515146
Date    : 18 September 2016
File    : controller.c v1.00
Desc    : File to support main app, processing state
*/

#include "boolean.h"
#include <stdio.h>
#include "model.h"
#include <string.h>

char * transisi(DFA *dfa, char * init, char sigmax){
  int idx = 0 ;
  boolean found = false;

  while (idx < (*dfa).countDelta && !found) {
    if (strcmp((*dfa).delta[idx].istate,init) == 0 && (*dfa).delta[idx].sigma == sigmax) {
      found = true;
    }else{
      idx++;
    }
  }

  return (*dfa).delta[idx].fstate;

}

char * getInitialState(DFA *dfa){
  return (*dfa).Start;
}

boolean isFinal(DFA dfa, char * state){
    boolean found = false;
    int idx = 0;
    while (idx < dfa.countFinal) {
```

```
    if ((dfa.Final[idx], state) == 0 ) {
      found = true;
    }else{
      idx++;
    }
  }

  return found;
}

boolean isAlphabetValid(DFA dfa, char x){
 boolean found = false;
 int idx = 0;
 while (idx < dfa.countAlphabet) {
  if (dfa.Alphabet[idx] == x ) {
   found = true;
  }else{
   idx++;
  }
 }

 return found;
}

void proses(DFA *dfa, char * kalimat){
 char CState[4];
 char * X = getInitialState(&(*dfa));
 int i =0;
 int j;
 for (i = 0; i < 4; i++) {
  CState[i] = X[i];
 }
 printf("\n    %s\n", CState);
 for (i = 0; i < strlen(kalimat); i++) {
  X = transisi(&(*dfa), CState, kalimat[i]);
  for (j = 0; j < 4; j++) {
   CState[j] = X[j];
  }
  printf("    %s\n", CState);
 }

 strcpy(X, CState);

}

char * prosesFinal(DFA *dfa, char * kalimat){
 char CState[4];
 char * X = getInitialState(&(*dfa));
 int i =0;
 int j;
 for (i = 0; i < 4; i++) {
```

```
    CState[i] = X[i];
  }
  for (i = 0; i < strlen(kalimat); i++) {
    X = transisi(&(*dfa), CState, kalimat[i]);
    for (j = 0; j < 4; j++) {
      CState[j] = X[j];
    }
  }

  strcpy(X, CState);

  return X;
}


boolean isAccepted(char * state){
  return (state[3] == 'a');
}
```

File : model.h

```
/*
Marble Toy Automata Machine
Author  : Fadhil Imam Kurnia - 13515146
Date    : 18 September 2016
File    : model.h v1.00
Desc    : File to support main app, connecting with data
*/

#ifndef MODEL_H
#define MODEL_H

#include "boolean.h"
#include <stdio.h>

typedef struct{
  char istate[20];
  char sigma;
  char fstate[20];
} DELTA;

typedef struct {
  char Alphabet[5];
  int countAlphabet;
  char *State[20];
  int countState;
  char Start[10];
  char *Final[20];
  int countFinal;
  DELTA delta[100];
  int countDelta;
} DFA;
```

```
void ReadFile(DFA *dfa);


#endif
```

File : controller.h

```
/*
Marble Toy Automata Machine
Author  : Fadhil Imam Kurnia - 13515146
Date    : 18 September 2016
File    : controller.h v1.00
Desc    : File to support main app, processing state
*/

#ifndef CONTROLLER_H
#define CONTROLLER_H

#include "boolean.h"
#include <stdio.h>
#include "model.h"

char * transisi(DFA *dfa, char * init, char sigmax);
char * getInitialState(DFA *dfa);
boolean isFinal(DFA dfa, char * state);
boolean isAlphabetValid(DFA dfa, char x);
void proses(DFA *dfa, char * kalimat);
char * prosesFinal(DFA *dfa, char * kalimat);
boolean isAccepted(char * state);

#endif
```

File view.c merupakan file yang digunakn untuk membuat antarmuka program dan tidak berisi algoritma sama sekali. File model.c digunakan untuk menghubungkan program dengan file eksternal dan menyimpan data dari file eksternal kedalam program. Sedangkan file controller.c digunakan untuk memproses data yang telah didapat dari file eksternal melalui model.c. Skema program ini menggunakan konsep MVC (*Model View Controller*) agar dapat lebih mudah dikembangkan kedepannya.

Berikut adalah file eksternal yang digunakan untuk memberi masukan program:

| File : language.dfa |
| --- |
| alphabet : {A,B}<br>state : {000r,000a,001a,010r,010a,011r,100r,100a,101r,101a,110r,110a,111r}<br>start : 000r<br>final : {000a,001a,010a,100a,101a,110a}<br>delta :<br>(000r\|A=100r)<br>(000r\|B=011r)<br>(000a\|A=100r)<br>(000a\|B=011r)<br>(001a\|A=101r)<br>(001a\|B=000a)<br>(010r\|A=110r)<br>(010r\|B=001a)<br>(010a\|A=110r)<br>(010a\|B=001a)<br>(011r\|A=111r)<br>(011r\|B=010a)<br>(100r\|A=010r)<br>(100r\|B=111r)<br>(100a\|A=010r)<br>(100a\|B=111r)<br>(101r\|A=011r)<br>(101r\|B=100a)<br>(101a\|A=011r)<br>(101a\|B=100a)<br>(110r\|A=000a)<br>(110r\|B=101a)<br>(110a\|A=0001)<br>(110a\|B=101a)<br>(111r\|A=001a)<br>(111r\|B=110a) |

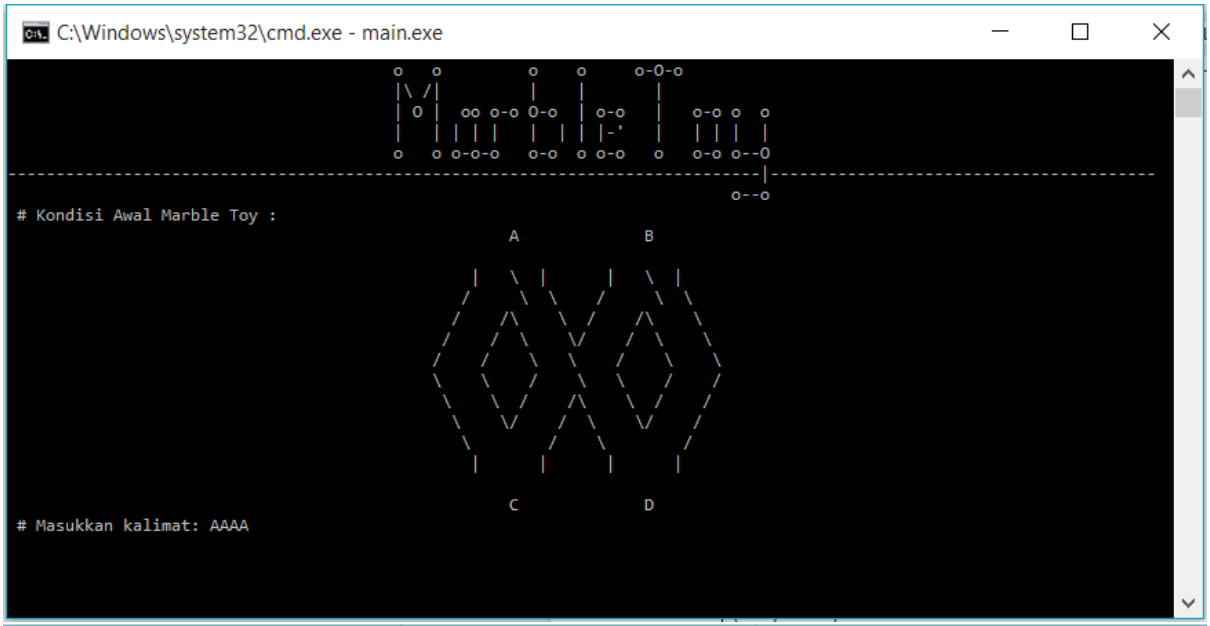
**D. Daftar Asumsi**

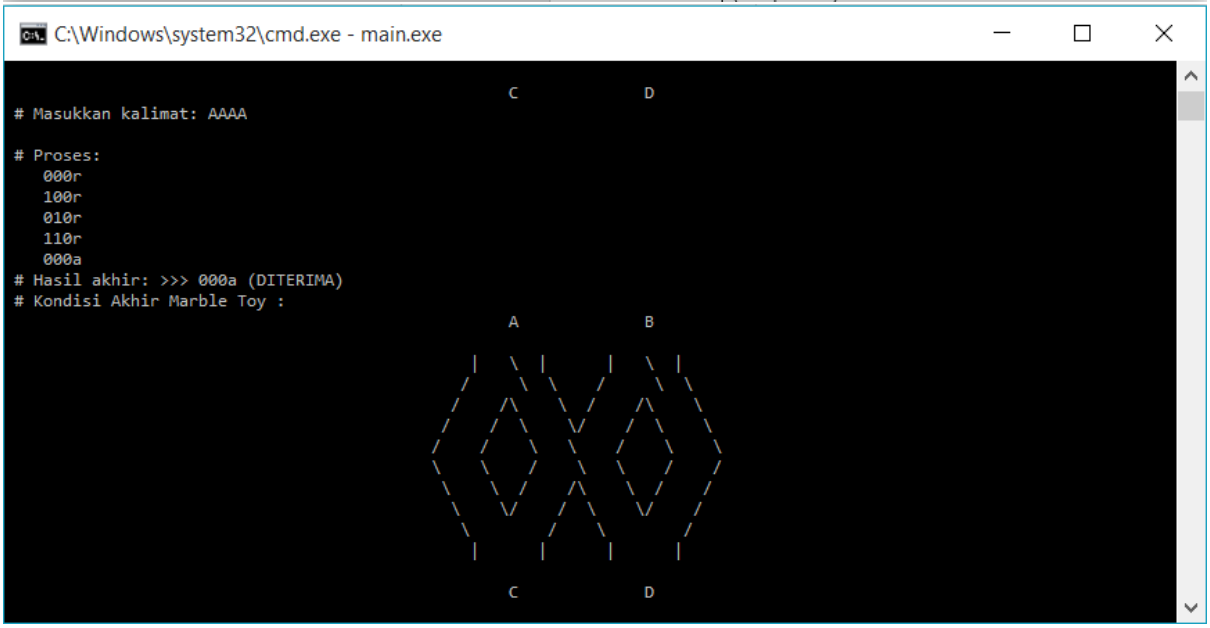
1. Diasumsikan masukan user selalu sesuai dengan alfabet A B.
2. Diasumsikan *state* yang mungkin terjadi adalah seperti yang telah dijelaskan pada Bab B.
3. Diasumsikan data dalam file eksternal sudah benar

## E. Contoh Masukan dan Keluaran

Contoh masukkan dan keluaran dapat dilihat pada gambar gambar berikut:

Kasus masukan AAAA



Kasus masukan AA