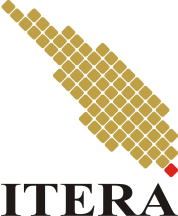
**LAPORAN TUGAS AKHIR**

**ALGORITMA DAN SRTUKTUR DATA**

Dosen pembimbing :

Rahman Indra Kesuma, S.Kom., M.Cs.

****

**GAME ZUMA**

**DISUSUN OLEH**

**Adila Gita Risnanda (14117047)**

**Febiola Agatha (14117120)**

**Mona Adhana (14117069)**

**Renhard Halim (14117104)**

**PROGRAM STUDI INFORMATIKA**

**JURUSAN SAINS**

**INSTITUT TEKNOLOGI SUMATERA**

**KABUPATEN LAMPUNG SELATAN**

**2018**

**HALAMAN PENGESAHAN**

Yang bertanda tangan dibawah ini :

Nama asisten : Agung Adi

Nim asisten : 14116104

Menyatakan nama-nama dibawah ini :

Kelompok 2

Ketua : Febiola Agatha

NIM : 14117120

Anggota

1. Adila Gita Risnanda (14117047)
2. Mona Adhana (14117069)
3. Renhard Halim (14117104)

Bahwa benar laporan dan program yang dibuat merupakan karya kelompok yang telah diperiksa dan disahkan.

Lampung Selatan,17 Desember 2018

Asisten Ketua Kelompok

Agung Adi Febiola Agatha 14116104 14117120

# KATA PENGANTAR

Puji syukur kami ucapkan kepada Tuhan Yang Maha Esa atas berkat dan rahmatnya penyusuna laporan tugas besar “ Algoritma dan Struktur Data ” dapat diselesaikan dengan baik. Laporan tugas besar ini dibuat dalam rangka memenuhi tugas algoritma dan struktur data.

Dalam pembuatan laporan ini, kami mengucapkan terimakasih kepada dosen pembimbing dan kepada asisten pembimbing yang telah membatu dalam pembuatan tugas besar ini.

Demikian laporan ini kami buat dengan segala kelebihan dan kekurangan. Oleh sebab itu, kritik dan saran yang membangun demi perbaikan laporan ini sangat kami harapkan. Semoga laporan ini dapat memberikan manfaat dan pengetahuan bagi pembaca.

Bandar Lampung,desember 2018

Penulis

**DAFTAR ISI**

Contents

[KATA PENGANTAR 3](#_Toc532809740)

[BAB I 5](#_Toc532809741)

[PENDAHULUAN 5](#_Toc532809742)

[Latar Belakang 5](#_Toc532809743)

[Batasan Masalah 5](#_Toc532809744)

[Manfaat Penelitian 6](#_Toc532809745)

[Tujuan penelitian 6](#_Toc532809746)

[Sistematika Penulisan 6](#_Toc532809747)

[BAB II METODOLOGI 7](#_Toc532809748)

[Analisis Pemrograman 7](#_Toc532809749)

[Penulisan Algoritma atau Flowchart 7](#_Toc532809750)

[BAB III 9](#_Toc532809751)

[PEMBAHASAN DAN HASIL PROGRAM 9](#_Toc532809752)

[Kekurangan dan Kelebihan Program 9](#_Toc532809753)

[Source Code 9](#_Toc532809754)

[Kesimpulan 25](#_Toc532809755)

[Saran 25](#_Toc532809756)

[BAB V 25](#_Toc532809757)

[BAGIAN AKHIR 25](#_Toc532809758)

[Daftar Pustaka 25](#_Toc532809759)

[Lampiran 25](#_Toc532809760)

# BAB I

# PENDAHULUAN

# Latar Belakang

Aplikasi permainan (game) sekarang ini sudah semakin berkembang. Hal ini dibuktikan melalui media yang di pakai untuk game yang semakin beragam. Sekarang ini game tidak hanya dapat dimainkan melalui computer saja, tetapi sudah banyak yang menggunakkan gadget sebagai media untuk memainkan game.

Zuma merupakan permainan puzzle video pencocokan-ubin yang di terbitkan oleh PopCap Games. Permainan ini dapat dimainkan secara gratis online di beberapa situs web, dan dapat di beli untuk sejumlah platform, termasuk PDA, ponsel, dan iPod. Game ini tidak memandang usia, permainan satu ini coco untuk anak-anak, remaja, serta dewasa atau orang tua.

Cara bermain game ini tergolong cukup mudah. Kita hanya diminta untuk menebak angka di lintasan yang mempunai kesamaan angka lebih dari 2 atau sesuai denngan angka yang ingin kita samakan dan menebaknya sampai angka yang sama menjadi 3. Semakin cepat kita menghabiskan angka yang ada dalam waktu yang singkat maka bonus yang di dapat pun semakin banyak.

# Batasan Masalah

Batasan masalah pada laporan ini adalah sebagai berikut :

1. Program permainan zuma menggunakkan bahasa pemrograman C++.
2. Game ini hanya di mainkan oleh satu orang.
3. Dalam game ini hanya terdiri dari 1 level.
4. Game ini mengharuskan user mengisi angka dengan menebak angka yang akan disamakan.

# Manfaat Penelitian

Dengan adanya pembuatan program ini mahasiswa dituntut menjadi lebih kreatif dan sekaligus menambah wawasan akan ilmu yang telah mereka terima selama ini. Dan menerapkannya dalam kehidupan nyata.

# Tujuan penelitian

1. Memenuhi tugas besar mata kuliah Algoritma Struktur Data
2. Mengembangkan kemampuan penelilti dalam bahasa pemrograman C++

# Sistematika Penulisan

Untuk memudahkan pembaca dalam memahami laporan dan pembahasan program ini maka kami membuat sistem penulisan laporan sebagai berikut:

**BAB I PENDAHULUAN** berisi latar belakang, batasan masalah, manfaat penelitian, tujuan penelitian dan sistematika penulisan.

**BAB II METODOLOGI** berisi analisis pemrograman dan penulisan algoritma atau flowchart.

**BAB III PEMBAHASAN DAN HASIL PROGRAM** berisi kekurangan & kelebihan program, source code dan kesimpulan.

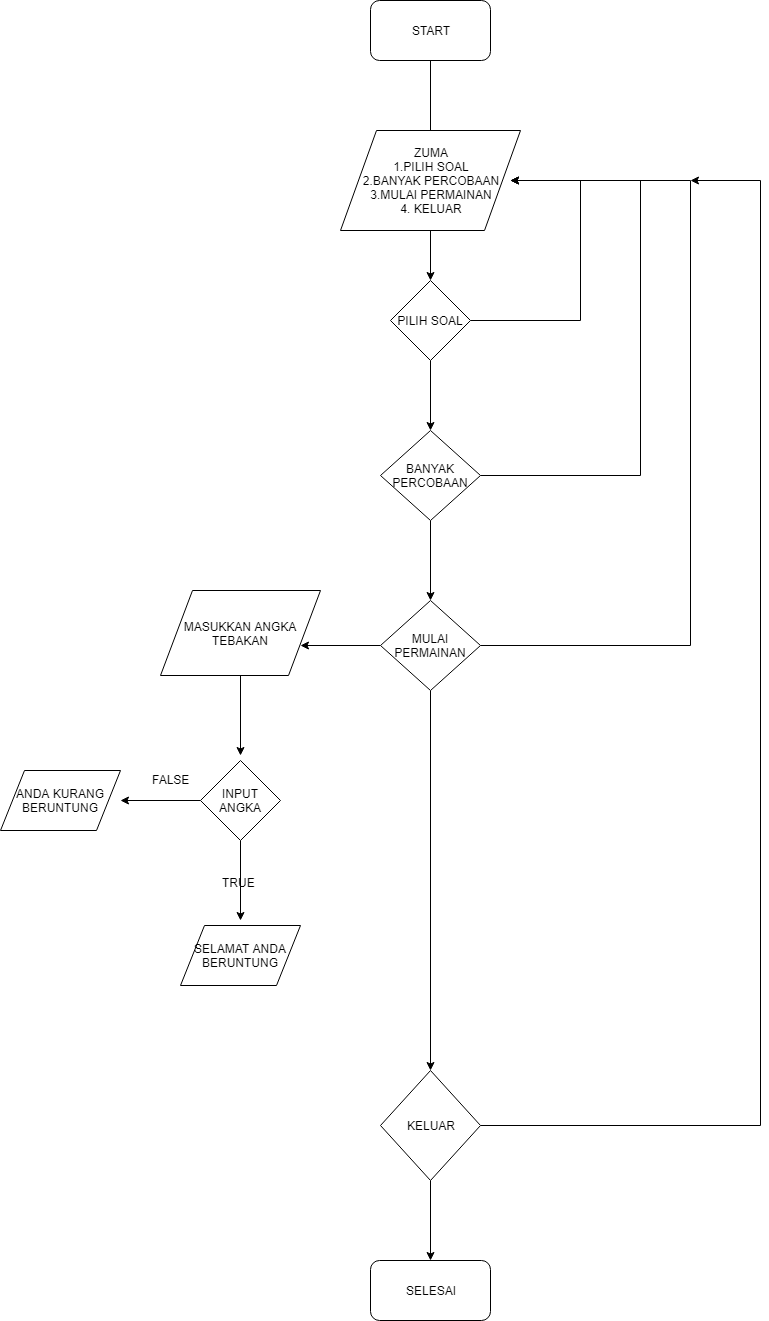
**BAB IV** berisi daftar pustaka dan lampiran.

# BAB II METODOLOGI

# Analisis Pemrograman

Program yang kami buat kali ini adalah sebuah representasi game ZUMA dimana pada game asli, user diharuskan menembakkan sebuah bola untuk mendapatkan 3 atau lebih bola yang berwarna sama. Namun program kami mengharuskan user menebak angka untuk mendapatkan 3 angka yg sama dengan cara mengisi soal yang telah kami sediakan. User dapat menginputkan nilai pada bagian yg bertanda (\_). User dapat menentukan sendiri kesempatan menebak angka atau dalam game ZUMA disebut nyawa. Angka yang dapat ditebak antara 0-9.

# Penulisan Algoritma atau Flowchart



# BAB III

# PEMBAHASAN DAN HASIL PROGRAM

# Kekurangan dan Kelebihan Program

1. Kelebihan Program

Dapat melatih pemain untuk berfikir lebih cepat dalam menentukan angka mana yang dijadikan target agar pemain bisa cepat menang dalam permainan.

1. Kekurangan Program

Kesempatan harus sampai 0 baru tulisan "anda belum beruntung" akan muncul.

Memasukan data harus berurut.

# Source Code

#include <iostream>

#include <stdlib.h>

#include <time.h>

#define max 10

#include <windows.h>

using namespace std;

typedef char infotype;

typedef struct TElmtList \*address;

typedef struct TElmtList{

infotype info;

address next;

address prev;

} ElmtList;

typedef struct {

address first;

address last;

}List ;

//SELEKTOR

#define Info(P) (P)->info

#define Next(P) (P)->next

#define Prev(P) (P)->prev

#define First(L) ((L).first)

#define Last(L) ((L).last)

//KONSTANTA

#define Nil NULL

#define Infinity 99999

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* TEST LIST KOSONG \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

bool IsEmpty(List L) {

/\* Mengirim true jika list kosong. Lihat definisi di atas. \*/

return (First(L) == NULL && Last(L) == NULL);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* PEMBUATAN LIST KOSONG \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void createList(List \*L) {

First(\*L) = NULL;

Last(\*L) = NULL;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Manajemen Memori \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

address Allocation(infotype x) {

/\* Mengirimkan address hasil alokasi sebuah elemen \*/

/\* Jika alokasi berhasil, maka address tidak nil. \*/

/\* Misalnya: menghasilkan P, maka Info(P)=X, Next(P)=Nil, Prev(P)=Nil \*/

/\* Jika alokasi gagal, mengirimkan Nil. \*/

address NewElmt;

NewElmt = (ElmtList\*) malloc (sizeof(ElmtList));

Info(NewElmt) = x;

Next(NewElmt) = Nil;

Prev(NewElmt) = Nil;

return NewElmt;

}

void Deallocation(address hapus) {

/\* I.S. P terdefinisi \*/

/\* F.S. P dikembalikan ke sistem \*/

/\* Melakukan dealokasi/pengembalian address P \*/

free(hapus);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* PENCARIAN SEBUAH ELEMEN LIST \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

address Search (List L, infotype X){

/\* Mencari apakah ada elemen list dengan Info(P)=X \*/

/\* Jika ada, mengirimkan address elemen tersebut. \*/

/\* Jika tidak ada, mengirimkan Nil \*/

address P = First(L);

while (P != Nil){

if (Info(P) == X){

return P;

}

P = Next(P);

}

return Nil;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* PRIMITIF BERDASARKAN NILAI \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\* PENAMBAHAN ELEMEN \*\*\*/

void InsertFirst(List \*L, infotype x) {

/\* I.S. L mungkin kosong \*/

/\* F.S. Melakukan alokasi sebuah elemen dan \*/

/\* menambahkan elemen pertama dengan nilai X jika alokasi berhasil \*/

address NewElmt;

NewElmt = Allocation(x);

if (NewElmt != NULL) {

if(IsEmpty(\*L)){

First(\*L) = NewElmt;

Last(\*L) = NewElmt;

} else{

Next(NewElmt) = First(\*L);

Prev(First(\*L)) = NewElmt;

First(\*L) = NewElmt;

}

}

}

void InsertLast(List \*L, infotype x) {

/\* I.S. L mungkin kosong \*/

/\* F.S. Melakukan alokasi sebuah elemen dan \*/

/\* menambahkan elemen list di akhir: elemen terakhir yang baru \*/

/\* bernilai X jika alokasi berhasil. Jika alokasi gagal: I.S.= F.S. \*/

address NewElmt;

NewElmt = Allocation(x);

if (NewElmt != NULL) {

if(IsEmpty(\*L)){

InsertFirst(L,x);

} else{

Next(Last(\*L)) = NewElmt;

Prev(NewElmt) = Last(\*L);

Last(\*L) = NewElmt;

}

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* PRIMITIF BERDASARKAN ALAMAT \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\* PENAMBAHAN ELEMEN BERDASARKAN ALAMAT \*\*\*/

void InsertFirst (List \*L, address P){

/\* I.S. Sembarang, P sudah dialokasi \*/

/\* F.S. Menambahkan elemen ber-address P sebagai elemen pertama \*/

if(IsEmpty(\*L)){

First(\*L) = P;

Last(\*L) = P;

} else{

Next(P) = First(\*L);

Prev(First(\*L)) = P;

First(\*L) = P;

}

}

void InsertLast (List \*L, address P){

/\* I.S. Sembarang, P sudah dialokasi \*/

/\* F.S. P ditambahkan sebagai elemen terakhir yang baru \*/

if(IsEmpty(\*L)){

InsertFirst(L,P);

} else{

Next(Last(\*L)) = P;

Prev(P) = Last(\*L);

Last(\*L) = P;

}

}

void InsertAfter (List \*L, address P, address Prec){

/\* I.S. Prec pastilah elemen list; P sudah dialokasi \*/

/\* F.S. Insert P sebagai elemen sesudah elemen beralamat Prec \*/

if(Next(Prec) == Nil){

InsertLast(L,P);

}else{

Next(P) = Next(Prec);

Prev(P) = Prec;

Prev(Next(Prec))= P;

Next(Prec) = P;

}

}

void InsertBefore (List \*L, address P, address Succ){

/\* I.S. Succ pastilah elemen list; P sudah dialokasi \*/

/\* F.S. Insert P sebagai elemen sebelum elemen beralamat Succ \*/

if(Succ == First(\*L)){

InsertFirst(L,P);

}else{

Next(Prev(Succ)) = P;

Prev(P) = Prev(Succ);

Next(P) = Succ;

Prev(Succ) = P;

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* PROSES SEMUA ELEMEN LIST \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void PrintForward (List L){

/\* I.S. List mungkin kosong \*/

/\* F.S. Jika list tidak kosong, isi list dicetak dari elemen pertama \*/

/\* ke elemen terakhir secara horizontal ke kanan: [e1,e2,...,en] \*/

/\* Contoh : jika ada tiga elemen bernilai 1, 20, 30 akan dicetak: [1,20,30] \*/

/\* Jika list kosong : menulis [] \*/

/\* Tidak ada tambahan karakter apa pun di awal, akhir, atau di tengah \*/

cout<<"[";

if(!IsEmpty(L)){

address temp =First(L);

while(Next(temp)!=NULL){

if(Info(temp) == 0){

cout<<"\_"<<" ";

}

else{

cout<<Info(temp)<<" ";

}

temp=Next(temp);

}

if(Info(temp) == 0){

cout<<"\_"<<" ";

}

else{

cout<<Info(temp);

}

}

cout<<"]"<<endl;

}

void konvert(int random, List \*L){

if(random == 1){

InsertLast(L, '1');

} else if(random == 2){

InsertLast(L, '2');

} else if(random == 3){

InsertLast(L, '3');

} else if(random == 4){

InsertLast(L, '4');

} else if(random == 5){

InsertLast(L, '5');

} else if(random == 6){

InsertLast(L, '6');

} else if(random == 7){

InsertLast(L, '7');

} else if(random == 8){

InsertLast(L, '8');

} else if(random == 9){

InsertLast(L, '9');

} else if(random == 0){

InsertLast(L, '0');

}

}

void random\_angka (List \*L){

int random,random2,k=0;

srand(time(0));

for(int i=0;i<10;i++){

random=rand()%9;

random2=rand()%4;

if(random2&&k<5){

InsertLast(L, '\_');

k++;

}else{

konvert(random, L);

}

}

}

void random\_angka2 (List \*L){

int random,random2,k=0;

//srand(time(0));

for(int i=0;i<10;i++){

random=rand()%9;

random2=rand()%4;

if(random2&&k<5){

InsertLast(L, '\_');

k++;

}else{

konvert(random, L);

}

}

}

address mencari (List L, infotype x,infotype y){

address P=First(L);

while (P!=NULL){

if(Prev(P)!=NULL&&Next(P)!=NULL&&Info(Next(P))==y&&Info(Prev(P))==x){

if(Info(P)=='\_'){

cout<<Info(P)<<endl;

return P;

}

}

P=Next(P);

}

return NULL;

}

bool cekawal(List L,address temp){

return (Info(temp) == Info(Next(First(L))) && Info(temp) == Info(Next(Next(First(L)))));

}

bool cekakhir(List L,address temp){

return (Info(temp) == Info(Prev(First(L))) && Info(temp) == Info(Prev(Prev(First(L)))));

}

bool cekPrev(List L,address temp){

address Prec = Prev(temp);

bool status = true;

if(temp!=Nil)

{

if(Prec!=Nil)

{

if(Prev(Prec)!=Nil)

{

if((Info(temp) == Info(Prec))&&(Info(temp) == Info(Prev(Prec))))

{

status = true;

}

else

{

status = false;

}

}

else

{

status = false;

}

}

else{

status = false;

}

}

else

{

status = false;

}

return (status);

}

bool cekNext(List L,address temp){

address Succ = Next(temp);

bool status = true;

if(temp!=Nil)

{

if(Succ!=Nil)

{

if(Next(Succ)!=Nil)

{

if((Info(temp) == Info(Succ))&&(Info(temp) == Info(Next(Succ))))

{

status = true;

}

else

{

status = false;

}

}

else

{

status = false;

}

}

else{

status = false;

}

}

else

{

status = false;

}

return (status);

}

bool cekBetween(List L,address temp){

address Prec = Prev(temp);

address Succ = Next(temp);

bool status = true;

if(temp!=Nil)

{

if(Prec!=Nil && Succ!=Nil)

{

if(Info(temp) == Info(Prec) && Info(temp) == Info(Succ))

{

status = true;

}

else

{

status = false;

}

}

else{

status = false;

}

}

else

{

status = false;

}

return (status);

}

bool cekList(List L,address temp){

if(cekPrev(L,temp) || cekNext(L,temp)){

return true;

}

else{

if(cekBetween(L,temp)){

return true;

}

else{

return false;

}

}

}

void menu(List \*L, List \*Q){

Menu:

cout<<endl;

int menu1,n,keluar,ksmptn;

cout<<"==================================================================================================="<<endl;

cout<<" Ketentuan Cara Bermain GAME ZUMA "<<endl;

cout<<" 1.Permainan ini harus Pilih soal terlebih dahulu"<<endl;

cout<<" 2.Sebelum bermain,tentukan banyak percobaan"<<endl;

cout<<" 3.Setelah melakukan step by step diatas, maka anda bisa bermain game zuma "<<endl;

cout<<"==================================================================================================="<<endl;

cout<<endl;

cout<<endl;

cout<<" Selamat Datang di Game ZUMA"<<endl;

cout<<" 1. Pilih Soal"<<endl;

cout<<" 2. Pilih Banyak Percobaan"<<endl;

cout<<" 3. Mulai Permainan"<<endl;

cout<<" 4. Keluar"<<endl;

cout<<" Pilih Menu : ";cin>>menu1;

switch(menu1){

case 1 :

random\_angka(L);

random\_angka2(Q);

cout<<" 1. "; PrintForward(\*L);

cout<<endl;

cout<<" 2. "; PrintForward(\*Q);

cout<<" Pilih Soal : ";

cin>>n;

if(n==2){

\*L=\*Q;

}

goto Menu;

break;

case 2 :

cout<<" Banyak Percobaan : ";cin>>ksmptn;

goto Menu;

break;

case 3 :

break;

case 4 :

cout<<endl;

cout<<" KELUAR DARI PERMAINAN ? "<<endl;

cout<<endl;

cout<<" 1. Keluar "<<endl;

cout<<" 2. Tidak "<<endl;

cout<<endl;

cout<<" keluar/tidak ? ";cin>>keluar;

if(keluar == 1){

cout<<endl;

cout<<" ANDA MENINGGALKAN PERMAINAN ";

exit (0);

}

if(keluar == 2){

goto Menu;

}

break;

}

while (ksmptn>0){

int i=0;

char pilih;

infotype tebak,tebakangka1,tebakangka2;

address temp;

PrintForward(\*L);

cout<<" Kesempatan anda bermain = "<<ksmptn<<endl;;

cout<<" Masukan Angka tebakan anda : ";

cin>>tebak;

cout<<" Posisi Angka Tebakan Anda : "<<endl;

cout<<" a. Sebelum "<<endl;

cout<<" b. Setelah "<<endl;

cout<<" c. Awal"<<endl;

cout<<" d. Akhir"<<endl;

cout<<" Masukkan Posisi : ";cin>>pilih;

switch (pilih){

case 'a':

cout<<" sebelum Angka : ";

cin>>tebakangka1;

cout<<" Setelah Angka : ";

cin>>tebakangka2;

temp=mencari (\*L,tebakangka2,tebakangka1);

if(temp!=NULL){

Info(temp)=tebak;

}

PrintForward(\*L);

if(temp!=NULL&&(cekPrev(\*L,temp) || cekBetween(\*L,temp) || cekNext(\*L,temp))){

cout<<"==========================================================================="<<endl;

cout<<"== SELAMAT ANDA BERUNTUNG =="<<endl;

cout<<"==========================================================================="<<endl;

cout<<endl;

exit(0);

break;

}

break;

case 'b':

cout<<" Setelah Angka : ";

cin>>tebakangka2;

cout<<" Sebelum Angka : ";

cin>>tebakangka1;

temp=mencari (\*L,tebakangka2,tebakangka1);

if(temp!=NULL){

Info(temp)=tebak;

}

PrintForward(\*L);

if(temp!=NULL&&(cekPrev(\*L,temp) || cekBetween(\*L,temp) || cekNext(\*L,temp))){

cout<<"==========================================================================="<<endl;

cout<<"== SELAMAT ANDA BERUNTUNG =="<<endl;

cout<<"==========================================================================="<<endl;

cout<<endl;

exit(0);

break;

}

break;

case 'c':

Info(First(\*L))=tebak;

temp=First(\*L);

PrintForward(\*L);

if(temp!=NULL&&cekawal(\*L,temp)){

cout<<"==========================================================================="<<endl;

cout<<"== SELAMAT ANDA BERUNTUNG =="<<endl;

cout<<"==========================================================================="<<endl;

cout<<endl;

exit(0);

break;

}

break;

case 'd':

Info(Last(\*L))=tebak;

temp=Last(\*L);

PrintForward(\*L);

if(temp!=NULL&&cekakhir(\*L,temp)){

cout<<"==========================================================================="<<endl;

cout<<"== SELAMAT ANDA BERUNTUNG =="<<endl;

cout<<"==========================================================================="<<endl;

cout<<endl;

exit(0);

break;

}

break;

}

ksmptn--;

}

if(ksmptn<1){

PrintForward(\*L);

cout<<endl;

cout<<endl;

cout<<"==========================================================================="<<endl;

cout<<"== ANDA KURANG BERUNTUNG == "<<endl;

cout<<"==========================================================================="<<endl;

}

}

int main(){

List L, Q;

createList(&L);

createList(&Q);

srand(unsigned(time(0)));

cout<<"==================================================================================================="<<endl;

cout<<"==================================================================================================="<<endl;

cout<<"==== =========== ==== ==== ==== === === ==== = ==== = ===="<<endl;

cout<<"==== ========== ==== ==== ==== === === ==== ==== ==== ===="<<endl;

cout<<"==== ==== ==== ==== ==== == == ==== ==== ==== ===="<<endl;

cout<<"==== ==== ==== ==== ==== == ==== ============== ===="<<endl;

cout<<"==== ========== ==== ==== ==== ==== ==== ==== ===="<<endl;

cout<<"==== =========== ========== ==== ==== ==== ==== ===="<<endl;

cout<<"==================================================================================================="<<endl;

cout<<"==================================================================================================="<<endl;

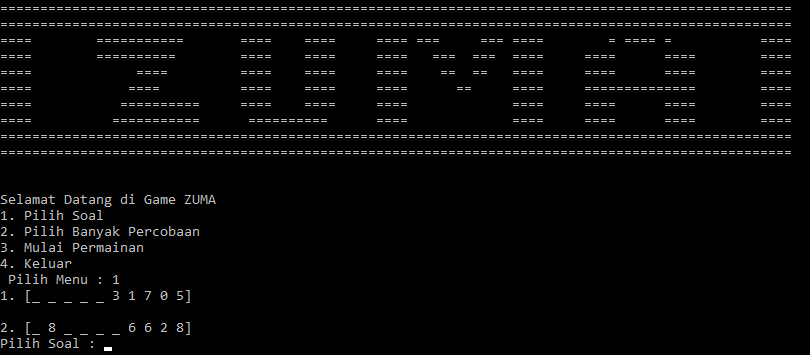
cout<<endl;

menu(&L, &Q);

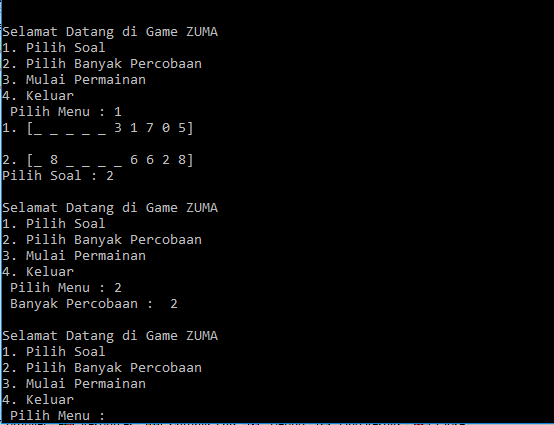
return 0;

}

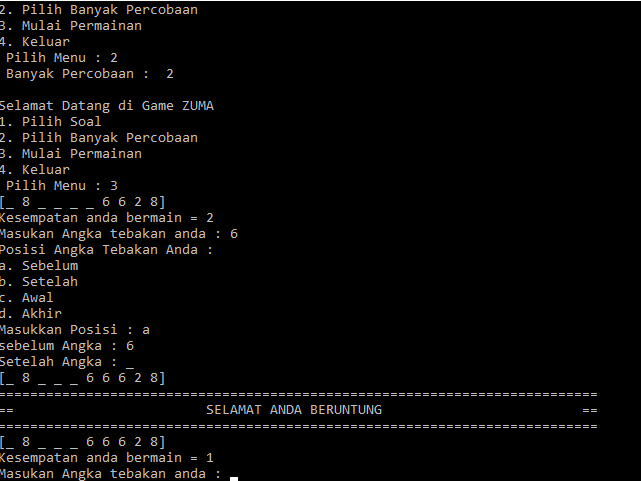
Tampilan awal game ZUMA



Setelah user memasukkan pilihan 1



Tampilan ketika user telah memilih soal dan memasukkan banyak kesempatan user dapat bermain



Permainan dapat dimulai setelah user menginputkan pilihan untuk memulai permainan.

# Kesimpulan

Program yang kami buat berjalan dengan lancar dan sesuai dengan representasi game ZUMA yang ingin kami tiru. Dalam penelitian kali ini kami menggunakan bahasa pemrograman C++. Permainan ini menantang pemain untuk berfikir lebih cerdas dan tangkas dalam menebak angka. Sekaligus mengasah kemampuan yang dimiliki oleh pembuat game dalam pemahamannya selama belajar Algoritma Struktur Data.

# Saran

# Program game zuma yang kami buat masih memiliki kekurangan dan di perlukan pengembangan lebih lanjut sesuai dengan kebutuhan user. Oleh karena itu, penulis menerima masukkan atau kritikan yang membangun demi terciptanya program yang lebih sempurna lagi.

# 

# BAB V

# BAGIAN AKHIR

# Daftar Pustaka

<https://www.anneahira.com/zuma-game.htm>  
<https://repository.widyatama.ac.id>

# Lampiran