**LAPORAN PRAKTIKUM ALGORITMA DAN PEMROGRAMAN**

**MODUL 4 – POINTER DAN FUNGSI**

****

Oleh :

Muhammad Firyanul Rizky 1708561006

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS UDAYANA**

**TAHUN 2018**

**BAB I**

**LANDASAN TEORI**

1.1. Pointer

1.1.1. Pengertian Pointer

Pointer (variabel penunjuk) adalah suatu variabel yang berisi alamat memori dari suatu variabel lain. Alamat ini merupakan lokasi dari obyek lain (biasanya variabel lain) di dalam memori. Contoh, jika sebuah variabel berisi alamat dari variabel lain, variable pertama dikatakan menunjuk ke variabel kedua. Operator Pointer ada dua, yaitu :

1. Operator &

* Operator & bersifat unary (hanya memerlukan satu operand saja).
* Operator & menghasilkan alamat dari operandnya.

2. Operator \*

* Operator \* bersifat unary (hanya memerlukan satu operand saja).
* Operator \* menghasilkan nilai yang berada pada sebuah alamat.

1.1.2. Deklarasi Pointer

Seperti halnya variabel yang lain, variabel pointer juga harus dideklarasikan terlebih dahulu sebelum digunakan. Bentuk Umum :

*Tipe\_data \*nama\_pointer;*

Tipe data pointer mendefinisikan tipe dari ob yek yang ditunjuk oleh pointer. Secara teknis, tipe apapun dari pointer dapat menun jukkan lokasi (dimanapun) dalam memori. Bahkan operasi pointer dapat dilaksanakan rel atif terhadap tipe dasar apapun yang ditunjuk. Contoh, ketika kita mendeklarasikan pointer dengan tipe int\*, compiler akan menganggap alamat yang ditunjuk menyimpan nilai integer - walaupun sebenarnya bukan (sebuah pointer int\* selalu menganggap bahwa ia menunjuk ke sebuah obyek bertipe integer, tidak peduli isi sebenarnya). Karenanya, sebelum mendeklarasikan sebuah pointer, pastikan tipenya sesuai dengan tipe obyek yang akan ditunjuk.

Contoh :

int \*px;

char \*sh;

1.1.3. Operasi Pointer

* Operasi Penugasan

Suatu variable pointer seperti halnya variable yang lain, juga bias mengalami operasi penugasan. Nilai dari suatu variable pointer dapat disalin ke variable pointer yang lain.

* Operasi Aritmatika

Suatu variabel pointer hanya dapat dila kukan operasi aritmatika dengan nilai integer saja. Operasi yang biasa dilakukan adalah operasi penambahan dan pengurangan. Operasi penambahan dengan suatu nilai menunju kkan lokasi data berikutnya (index selanjutnya) dalam memori. Begitu juga operasi pengurangan.

* Operasi Logika

Suatu pointer juga dapat dikenai operasi logika.

1.1.4. Pointer dan String

#include “stdio.h”

#include “conio.h”

char \*nama1 = “SPIDERMAN”;

char \*nama2 = “GATOTKACA”;

void main(){

char namax;

printf(“SEMULA :”);

printf(“Saya suka >> %s\n”, nama1);

printf(“Tapi saya juga suka >> %s\n”, nama2);

// Penukaran string yang ditunjuk oleh pointer nama1 dan nama2 printf(“SEKARANG :”);

printf(“Saya suka >> %s\n”, nama1);

printf(“Dan saya juga masih suka >> %s\n”, nama2);

getch();

}

1.1.5. Pointer Menunjuk Suatu Array

#include “stdio.h”

#include “conio.h”

void main(){

static int tgl\_lahir[] = { 13,9,1982 };

int \*ptgl;

ptgl = tgl\_lahir; // ptgl berisi alamat array printf(“Diakses dengan pointer”);

printf(“Tanggal = %i\n”, \*ptgl);

printf(“Bulan = %i\n”, \*(ptgl + 1));

printf(“Tahun = %i\n”, \*(ptgl + 2));

printf(“\nDiakses dengan array biasa\n”);

printf(“Tanggal = %i\n”, tgl\_lahir[0]);

printf(“Bulan = %i\n”, tgl\_lahir[1]);

printf(“Tahun = %i\n”, tgl\_lahir[2]);

getch();

}

1.1.6. Memberi Nilai Array dengan Pointer

#include “stdio.h”

#include “conio.h”

void main(){

int x[5], \*p, k;

p = x;

x[0] = 5; // x[0] diisi dengan 5 sehingga x[0] = 5

x[1] = x[0]; // x[1] diisi dengan x[0] sehingga x[1] = 5

x[2] = \*p + 2; // x[2] diisi dengan x[0] + 2 sehingga x[2] = 7

x[3] = \*(p+1) – 3; // x[3] diisi dengan x[1] - 3 sehingga x[3] = 2 x[4] = \*(x + 2); // x[4] diisi dengan x[2] sehingga x[4] = 7

for(k=0; k<5; k++){

printf(“x[%i] = %i\n”, k, x[k]);

}

getch();

}

1.2. Fungsi

Fungsi merupakan suatu bagian dari program yang dimaksudkan untuk mengerjakan suatu tugas tertentu dan letaknya terpisah dari program yang memanggilnya. Fungsi merupakan elemen utama dalam bahasa C karena bahasa C sendiri terbentuk dari kumpulan fungsi-fungsi. Dalam setiap program bahasa C, minimal terdapat satu fungsi yaitu fungsi main(). Fungsi banyak diterapkan dalam program program C yang terstruktur. Keuntungan penggunaan fungsi dalam program yaitu program akan memiliki struktur yang jelas dan juga akan menghindari penulisan bagian program yang sama.

Dalam bahasa C fungsi dapat dibagi menjadi dua, yaitu fungsi pustaka atau fungsi yang telah tersedia dalam Bahasa Pemrograman C dan fungsi yang didefinisikan atau dibuat oleh programmer.

1.2.1. Beberapa Fungsi Pustaka pada Bahasa Pemrograman C

Fungsi Operasi String (tersimpan dalam header file “string.h”)

* strcpy()
* Berfungsi untuk menyalin suatu string asal ke variable string tujuan.
* Bentuk umum : strcpy(var\_tujuan, string\_asal);
* strlen()
* Berfungsi untuk memperoleh jumlah karakter dari suatu string.
* Bentuk umum : strlen(string);
* strcat()
* Digunakan untuk menambahkan string sumber ke bagian akhir dari string tujuan.
* Bentuk umum : strcat(tujuan, sumber);
* strupr()
* Digunakan untuk mengubah setiap huruf dari suatu string menjadi huruf capital.
* Bentuk umum : strupr(string);
* strlwr()
* Digunakan untuk mengubah setiap huruf dari suatu string menjadi huruf kecil semua.
* Bentuk umum : strlwr(string);
* strcmp()
* Digunakan untuk membandingkan dua buah string.
* Hasil dari fungsi ini bertipe integer dengan nilai :

1. Negative, jika string pertama kurang dari string k edua.
2. Nol, jika string pertama sama dengan string kedua
3. Positif, jika string pertama lebih besar dari string kedua.

* Bentuk umum : strcmp(string1, string2);

Fungsi Operasi Karakter (tersimpan dalam header “ctype.h”)

* islower()
* Fungsi akan menghasilkan nilai benar (bukan nol) jika karakter merupakan huruf kecil.
* Bentuk umum : islower(char);
* isupper()
* Fungsi akan menghasilkan nilai benar (bukan nol) jika karakter merupakan huruf kapital.
* Bentuk umum : isupper(char);
* isdigit()
* Fungsi akan menghasilkan nilai benar (bukan nol) jika karakter merupakan sebuah digit.
* Bentuk umum : isdigit(char);
* tolower()
* Fungsi akan mengubah huruf capital menjadi huruf kecil.
* Bentuk umum : tolower(char);
* toupper()
* Fungsi akan mengubah huruf kecil menjadi huruf kapital.
* Bentuk umum : toupper(char);

Fungsi Operasi Matematik (tersimpan dalam header “math.h” dan “stdlib.h”)

* sqrt()
* Digunakan untuk menghitung akar dari sebuah bilangan.
* Bentuk umum : sqrt(bilangan);
* pow()
* Digunakan untuk menghitung pemangkatan suatu b ilangan.
* Bentuk umum : pow(bilangan, pangkat);
* sin(), cos(), tan()
* Masing-masing digunakan untuk menghitung nilai sinus, cosinus dan tangens dari suatu sudut.
* Bentuk umum : sin(sudut); cos(sudut); tan(sudut);
* atof()
  + Digunakan untuk mengkonversi nilai string menjadi bilangan bertipe double.
  + Bentuk umum : atof(char x);
* atoi()
* Digunakan untuk mengkonversi nilai string menjadi bilangan bertipe integer.
* Bentuk umum : atoi(char x);
* div()
  + Digunakan untuk menghitung hasil pembagian dan sisa pembagian.
  + Bentuk umum : div\_t div(int x, int y)
  + Strukturnya :

typedef struct{

int qout; // hasil pembagian int rem // sisa pembagian

} div\_t;

* max()
* Digunakan untuk menentukan nilai maksimal dari dua buah bilangan.
* Bentuk umum : max(bilangan1, bilangan2);
* min()
  + Digunakan untuk menentukan bilangan terkecil dari dua buah bilangan.
  + Bentuk umum : min(bilangan1, bilangan2);

1.2.2. Membuat Fungsi Sendiri

Deklarasi Fungsi

Sebelum digunakan (dipanggil), suatu fungsi harus dideklarasikan dan didefinisikan terlebih dahulu. Bentuk umum pendeklarasian fungsi adalah :

tipe\_fungsi nama\_fungsi(parameter\_fungsi);

Sedangkan bentuk umum pendefinisian fungsi adalah :

Tipe\_fungsi nama\_fungsi(parameter\_fungsi){

statement statement

...

...

}

Hal-hal yang perlu diperhatikan dalam penggunaan fungsi :

* Kalau tipe fungsi tidak disebutkan, maka akan dianggap sebagai fungis dengan nilai keluaran bertipe integer.
* Untuk fungsi yang memiliki keluaran bertipe bukan integer, maka diperlukan pendefinisian penentu tipe fungsi.
* Untuk fungsi yang tidak mempunyai nilai keluaran maka dimasukkan ke dalam tipe void
* Pernyataan yang diberikan untuk memberikan nilai akhir fun gsi berupa pernyataan return.
* Suatu fungsi dapat menghasilkan nilai balik bagi fungsi pemanggilnya.

Parameter Formal dan Parameter Aktual

* Parameter Formal adalah variabel yang ada pada daftar parameter dalam definisi fungsi.
* Parameter Aktual adalah variabel (parameter) yang dipakai dalam pemanggilan fungsi.

Dalam contoh program pertambahan di atas parameter formal terdapat pada pendefinisisan fungsi :

float tambah(float x, float y) //parameter formal

{

return (a+b);

}

Sedangkan parameter aktual terd apat pada pemanggilan fungsi :

void main(){

...

c = tambah(a, b); //parameter aktual

...

}

Cara Melewatkan Parameter

Cara melewatkan suatu parameter dalam Bahasa C ada dua cara yaitu :

1. Pemanggilan Secara Nilai (Call by Value)

* Call by value akan menyalin nilai dari parameter aktual ke parameter formal.
* Yang dikirimkan ke fungsi adalah nilai dari datanya, bukan alamat mem ori letak dari datanya.
* Fungsi yang menerima kiriman nilai akan menyimpannya di alamat terpisah dari nilai aslinya yang digunakan oleh bagian program yang memanggil fungsi.
* Perubahan nilai di fungsi (parameter formal) t idak akan merubah nilai asli di bagian program yang memanggilnya.
* Pengiriman parameter secara nilai adalah pengiriman searah, yaitu dari bagian program yang memanggil fungsi ke fungsi yang dipanggil.
* Pengiriman suatu nilai dapat dilakukan untuk suatu ungkapan, tidak hanya untuk sebuah variabel, elemen array atau konstanta saja.

2. Pemanggilan Secara Referensi (Call by Reference)

* Pemanggilan secara Referensi merupakan upaya untuk melewatkan alamat dari suatu variabel ke dalam fungsi.
* Yang dikirimkan ke fungsi adalah alamat letak dari nilai datanya, bukan nilai datanya.
* Fungsi yang menerima kiriman alamat ini makan menggunakan alamat yang sama untuk mendapatkan nilai datanya.
* Perubahan nilai di fungsi akan merubah nilai asli di bagian program yang memanggil fungsi.
* Pengiriman parameter secara referensi adalah pengiriman dua arah, yaitu dari fungsi pemanggil ke fungsi yang dipanggil dan juga sebaliknya.
* Pengiriman secara acuan tidak dapat bdilakukan untuk suatu ungkapan.

Penggolongan Variabel berdasarkan Kelas Penyimpanan (Storage Class)

1. Variabel lokal

* Variabel lokal adalah variabel yang dideklarasikan di dalam fungsi. Sifat-sifat variabel lokal :
* Secara otomatis akan diciptakan ketika fungsi dipanggil dan akan lenyap ketika proses eksekusi terhadap fungsi berakhir.
* Hanya dikenal oleh fungsi tempat variabel dideklarasikan
* Tidak ada inisialisasi secara otomatis (saat variabel diciptakan nilainya random).
* Dideklarasikan dengan menambahkan kata “ auto” (opsional).

2.Variabel global (eksternal)

Variabel global (eksternal) adalah variabel yang dideklarasikan di luar fungsi. Sifat- sifat variabel global :

* Dikenal (dapat diakses) oleh semua fungsi.
* Jika tidak diberi nilai awal secara otomatis berisi nilai nol.
* Dideklarasikan dengan menambahkan kata “ extern” (opsional).

3. Variabel Statis

* Variabel statis adalah variabel yang nilainya tetap dan bisa berupa variabel local
* (internal) dan variabel global (eksternal). Sifat-sifat variabel statis :
* Jika bersifat internal (lokal), maka variabel hanya dike nal oleh fungsi tempat variabel dideklarasikan.
* Jika bersifat eksternal (global), maka variabel dapat dipergunakan oleh semua fungsi yang terletak pada program yang sama.
* Nilai variabel statis tidak akan hilang walau eksekusi terhadap fungsi telah berakhir.
* Inisialisasi hanya perlu dilakukan sekali saja, yaitu pada saat fungsi dipanggil pertama kali.
* Jika tidak diberi nilai awal secara otomatis berisi nilai nol.
* Dideklarasikan dengan menambahkan kata “ static”.

4. Variabel Register

Variabel Register adalah variabel yang nilainya disimpan dalam re gister dan bukan dalam memori RAM. Sifat-sifat variabel register :

* Hanya dapat diterapkan pada variabel lokal yang bertipe int dan char.
* Digunakan untuk mengendalikan proses perulangan (looping).
* Proses perulangan akan lebih cepat karena variabel register memiliki kecepatan yang lebih tinggi dibandingkan variabel biasa.
* Dideklarasikan dengan menambahkan kata “ register”.

Fungsi Rekursif

Fungsi rekursif adalah fungsi yang memanggil dirinya sendiri.

#include “stdio.h”

#include “conio.h”

long int faktorial(int N); // prototype fungsi faktorial

void main(){

int N;

printf(“Berapa factorial ? “); scanf(“%i”, &N);

printf(“Faktorial dari %i = %ld\n”, N, faktorial(N));

getch();

}

// definisi fungsi factorial long int faktorial(int N) {

if(N==0) return(1);

else{

}

// memanggil fungsi faktorial()fungsi faktorial()

return(N \* faktorial(N - 1));

}

**BAB II**

**PERMASALAHAN**

**2.1 Studi Kasus**

Mr. X adalah seorang mata-mata di kepolisian. Kali ini dia mendapat tugas untuk memata-matai gembong narkoba kelas kakap. Setiap hari Mr.X harus melaporkan kegiatan gembong yang diintainya kepada atasannya dalam bentuk SMS (Short Message Service) yang sisandikan. Untuk memastikan bahwa pesan yang disampaikan benar-benar dari Mr.X, maka atasan Mr.X membuat aturan penulisan pesan yang harus digunakan oleh Mr.X pada setiap pengiriman pesannya. Mr.X hanya boleh mengirimkan 1 buah pesan setiap harinya, jika lebih dari itu maka operasi ini telah terbongkar dan ada kemungkinan Mr.X telah ditangkap. Aturan penulisan pesan yang digunakan adalah setiap pesan yang dikirim harus sepanjang 160 karakter. Penyusunan pesan tersebut dilakukan membentuk matriks spiral yang dimulai pusat matriks 1 karakter pertama, lalu 1 karakter berikutnya ke kanan, lalu 1 karakter berikutnya ke bawah, lalu 2 karakter berikutnya ke kiri, lalu 2 karakter berikutnya ke atas, 3 karakter berikutnya ke kanan, 3 karakter berikutnya ke bawah, dan seterusnya hingga semua karakter dalam kalimat termasuk dalam spiral. Khususnya, karakter spasi di ganti dengan “\_” (underscore), dan jika ada baris/kolom tersisa setelah karakter terakhir maka elemen-elemen matriks diisi juga dengan “\_” (underscore) tersebut. Misalnya kalimat “**Hari ini mereka mengirim barang dengan mobil pick-up pukul 12.10 ke jalan nirmala no.22, pukul 15.30 ke jalan asoka, dan pukul 18.02 ke jalan mahendradata**”

Dikodekan kedalam matriks sebagai berikut:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| \_ | \_ | \_ | \_ |  |  |  |  |  |  |  |  |  |
| \_ | a | s | o | k | a | , | \_ | d | A | n | \_ | p |
| \_ | \_ | n | \_ | n | i | R | m | a | L | a | \_ | u |
| A | n | a | i | l | \_ | P | i | c | K | - | n | k |
| T | a | l | b | i | r | I | m | \_ | B | u | o | u |
| A | l | a | o | g | n | I | \_ | m | A | p | . | l |
| D | a | j | m | n | i | H | a | e | R | \_ | 2 | \_ |
| A | j | \_ | \_ | e | \_ | I | r | r | A | p | 2 | 1 |
| R | \_ | e | n | m | \_ | A | k | e | N | u | , | 8 |
| D | e | k | a | g | n | E | d | \_ | G | k | \_ | . |
| N | k | \_ | 0 | 1 | . | 2 | 1 | \_ | L | u | p | 0 |
| E | \_ | 0 | 3 | . | 5 | 1 | \_ | l | U | k | u | 2 |
| H | a | m | \_ | n | a | L | a | j | \_ | e | k | \_ |

\_ \_ \_ \_ \_ a s o k a , \_ d a n \_ p \_ \_ n \_ n i r

m a l a \_ u a n a i l \_ p i c k – n k t a l b i

r i m \_ b u o u a l a o g n I \_ m a p . l d a j

m n i H a e r \_ 2 \_ a j \_ \_ e \_ i r r a p 2 1 r

\_ e n m \_ a k e n u , 8 d e k a g n e d \_ g k \_

. n k \_ 0 1 . 2 1 \_ l u p 0 e \_ 0 3 . 5 1 \_ l u

k u 2 h a m \_ n a l a j \_ e k \_

Untuk memudahkan Mr.X mengirim pesan kepada atasannya, bantulah Mr.X dengan sebuah program yang dapat mengubah pesan yang dikirmkan ke dalam bentuk di atas.

[input]

Hari ini mereka mengirim barang dengan mobil

pick-up pukul 12.10 ke jalan nirmala no.22,

pukul 15.30 ke jalan asoka, dan pukul 18.02 ke

jalan mahendradata

Max. 160 karakter

\_ \_ \_ \_ \_ a s o k a , \_ d a n \_ p \_ \_ n \_ n i r

m a l a \_ u a n a i l \_ p i c k – n k t a l b i

r i m \_ b u o u a l a o g n I \_ m a p . l d a j

m n i H a e r \_ 2 \_ a j \_ \_ e \_ i r r a p 2 1 r

\_ e n m \_ a k e n u , 8 d e k a g n e d \_ g k \_

. n k \_ 0 1 . 2 1 \_ l u p 0 e \_ 0 3 . 5 1 \_ l u

k u 2 h a m \_ n a l a j \_ e k \_

**BAB III**

**PEMBAHASAN**

Listing Kode Bahasa Pemrograman C :

#include <stdio.h>

#define MAX\_PESAN 160

char matrix [13][13],pesan[MAX\_PESAN],\*ppesan;

int jum\_pesan,jum\_matrix,baris,kolom=0;

int i,j,batas\_kanan,batas\_kiri,batas\_atas,batas\_bawah;

int main()

{

printf("\nMasukan pesan yang akan disandikan:\n\n\t");

gets(pesan);

ppesan=&pesan;

jum\_pesan=strlen(pesan);

if(jum\_pesan>MAX\_PESAN)

{

printf("\n\nPesan yang dimasukan terlalu panjang! Silahkan ulangi lagi\n\n");

main();

}

else if (jum\_pesan==0)

{

printf("Tidak ada pesan yang anda masukan, silahkan ulangi lagi!\n\n");

main();

}

for(i=0;i<7;i++)

{

if(i==0)

{

baris=kolom=6;

isi\_matrix(0,"tengah");

}

else

{

batas\_kanan = batas\_bawah = 6+i;

batas\_kiri = batas\_atas = 6-i;

baris=7-i;

kolom=batas\_kanan;

isi\_matrix(batas\_bawah,"bawah");

isi\_matrix(batas\_kiri,"kiri");

isi\_matrix(batas\_atas,"atas");

isi\_matrix(batas\_kanan,"kanan");

}

}

printf("\n\n\n=============================Matrix Enkripsi===============================\n\n");

for(i=0;i<13;i++)

{

printf(" ");

for(j=0;j<13;j++)

{

printf("%c ",matrix[i][j]);

}

printf("\n\n");

}

printf("\n\n\n============================================== =============================\n\n");

printf("Hasil yang telah disandikan : \n");

for(i=0;i<13;i++)

{

for(j=0;j<13;j++)

{

if(matrix[i][j]!= ' ')

printf("%c",matrix[i][j]);

}

}

}

void isi\_matrix(int batas,char arah[5])

{

while(0==0)

{

if(jum\_matrix<=jum\_pesan)

{

if(\*ppesan== ' '||\*ppesan == '\0')

matrix[baris][kolom]= '\_';

else

matrix[baris][kolom]=\*ppesan;

ppesan++;

}

else

{

if(jum\_matrix<MAX\_PESAN)

matrix[baris][kolom]= '\_';

else

matrix[baris][kolom]= ' ';

}

jum\_matrix++;

if(arah=="tengah")break;

else if(arah=="bawah")

{

baris++;

if(baris>batas)

{

baris--;

kolom--;

break;

}

}

else if(arah=="kiri")

{

kolom--;

if(kolom<batas)

{

baris--;

kolom++;

break;

}

}

else if(arah=="atas")

{

baris--;

if(baris<batas)

{

baris++;

kolom++;

break;

}

}

else if(arah=="kanan")

{

kolom++;

if(kolom>batas)

{

break;

}

}

}

}

Output:



Penjelasan:

Penjelasan Program diatas, pertama kita masukan library utama serta header tambahan yaitu #define MAX\_PESAN 160 yang nantinya digunakan untuk menentukan batasan karakter yang dimasukan dimana jumlah maksimal karakter yang digunakan yaitu 160 karakter. Kemudian deklarasikan beberapa variabel dimana variabel tersebut bertipe integer serta char. Kemudian buat program utama dimana pertama buat fungsi untuk menginputkan pesan yang kemudian pesan tersebut diperiksa oleh program. Apabila jumlah karakter yang dimasukan lebih dari 160 karakter atau tidak memasukan sama sekali maka program tidak akan mengeksekusi pesan tersebut dan meminta user untuk menginputkan ulang pesan. Dan apabila sebaliknya maka program tersebut akan melakukan looping untuk membentuk matriks yang berisi karakter yang dimulai dari tengah lalu ke bawah,

Kemudian ke kiri, selanjutnya ke atas, dan terakhir ke kanan hingga semua karakter telah membentuk matriks dimana program untuk menentukan arah dibuat secara terpisah dan untuk menggunakannya perlu fungsi pemanggilan.Kemudian untuk menampilkan hasil penyandian maka program melakukan looping untuk membaca pesan yang telah berbentuk matriks yang dibaca per baris.

Setelah program utama selesai dibuat, langkah selanjutnya adalah membuat program untuk menentukan arah karakter dalam matriks. Langkah pertama yaitu membuat baris program untuk menentukan apakah panjang karakter sesuai dengan batas maksimal inputan karakter atau tidak. Setelah itu membuat program untuk menentukan arah dimana karakter dimasukan dari tengah kemudian ke bawah, lalu ke kiri, lalu ke atas, dan ke kanan secara terus menerus hingga semua karakter telah diinputkan.