MODUL III

ARRAY

3.1 TUJUAN PRAKTIKUM

Tujuan Praktikum pada Modul III *Array* adalah:

1. Mahasiswa bisa memahami esensi dari *array* dan penggunaannya
2. Mahasiswa bisa mengimplementasikan penggunaan *array* ke dalam program komputer.
3. Mahasiswa bisa menggunakan fungsi manipulasi *string* ke dalam program komputer

3.2 DASAR TEORI

3.2.1 Pengertian Array

Variabel Larik atau lebih dikenal dengan *array* adalah tipe terstruktur yang terdiri dari sejumlah komponen-komponen data yang bertipe sama dalam urutan tertentu yang menggunakan nama yang sama. Suatu *array* mempunyai jumlah komponen yang banyaknya tetap. Banyaknya komponen dalam suatu larik ditunjukan oleh suatu indeks untuk membedakan variabel yang satu dengan variabel yang lainnya. Untuk menentukan ukuran *array*, perlu ditempatkan jumlah nilai yang dapat disimpan *array* dalam sebuah tanda kurung kurawal siku yang terletak sesuah nama *array* "[..]". Deklarasi berikut sebagai contoh untuk menciptakan *array* berbentuk secara umum (Warnilah, 2015):

```
Tipe data_Nama variabel[Jumlah element];
```

Berikut beberapa contoh deklarasi penerapan *array* dalam sebuah kode program sebagai berikut:

```
#include
Using namespace std;
Int main (){
Int Tab jumlah Hari [12];
Float Tab Nilai [15];
Char Tab Huruf [100];
String Tab Kata [100];
Long int Tab Nilai [150];
Point Tab Titik [20]; }
```

3.2.2 Jenis-Jenis Array

1. *Array* satu dimensi *Array* ini merupakan tipe *array* sering digunakan. *Array* ini memiliki satu subscript atau larik didalamnya. Adapun bentuk umum pendeklarasiannya adalah sebagai berikut (Harumy, 2016):

```
Tipe data_Nama variabel[Jumlah element];
```

2. *Array* dua dimensi *Array* dua dimensi merupakan *array* yang memiliki dua subscript. *Array* ini biasanya digunakan untuk membuat matriks. Indeks larik secara *default* dimulai

dari 0,0 dengan jumlah elemennya adalah indeks 1 x indeks 2. Adapun bentuk umum pendeklarasiannya adalah sebagai berikut (Harumy, 2016):

```
Tipe data_Nama variabel[Jumlah element1][Jumlah element2];
```

3. *Array* multidimensi Merupakan *array* yang memiliki lebih dari dua transkrip, seperti untuk menyebut *array* tiga dimensi, empat dimensi, lima dimensi dan seterusnya. Indeks larik secara default dimulai dari 0,0 dengan jumlah elemennya adalah indeks1 x indeks 2. x...indeks n. Bentuk umum pendeklarasiannya adalah sebagai berikut (Harumy, 2016):

```
Tipe data_Nama variabel[Jumlah element1][Jumlah element2]… [Jumlah
elementn];
```

3.2.3 String

String merupakan sebuah bentuk data yang sering dipakai dalam bahasa pemrograman untuk keperluan menampung dan memanipulasi data teks. *String* adalah suatu nilai karakter yang berurutan dan disimpan dalam suatu lokasi memori yang selalu diakhiri dengan karakter null. *String* dapat berupa huruf, angka, karakter khusus, maupun karakter Unicode (Suarga, 2012).

3.2.4 Fungsi String

Dalam bahasa C++ terdapat beberapa fungsi yang digunakan untuk operasi pengolahan *string*, sebagai berikut (Bachtiar, 2017):

1. Fungsi Strcat ( )

Fungsi strcat digunakan untuk menambahkan *string* sumber kebagian akhir dari *string* tujuan. Bentuk penulisannya sebagai berikut:

```
strcat (tujuan, sumber);
```

2. Fungsi Strcmp ( )

Fungsi strcmp digunakan untuk membandingkan *string* pertama dengan *string* kedua. Bentuk penulisannya sebagai berikut:

```
strcmp (str1, str2);
```

3. Fungsi Strcpy ( )

Fungsi strcpy digunakan untuk menyalin *string* asal ke variabel *string* tujuan, dengan syarat *string* tujuan harus mempunyai tipe data dan ukuran yang sama dengan *string* asal. Bentuk penulisannya sebagai berikut:

```
strcpy (tujuan, sumber);
```

4. Fungsi Strlen ( )

Fungsi strlen digunakan untuk memperoleh banyaknya karakter dalam *string*. File header yang harus disertakan adalah: "*string*.h". Bentuk penulisannya sebagai berikut:

```
strlen (str);
```

5. Fungsi Strrev ( )

Fungsi strrev digunakan untuk membalik letak urutan paling akhir dipindah ke urutan paling depan dan seterusnya. *File header* yang harus disiapkan adalah: "*string*.h". Bentuk penulisannya sebagai berikut:

```
strrev (str);
```

6. Fungsi Strupr ( )

Fungsi strupr digunakan untuk mengubah setiap huruf kecil dalam *string* menjadi huruf kapital. Bentuk penulisannya sebagai berikut:

```
strupr (str);
```

7. Fungsi Strlwr ( )

Fungsi strlwr digunakan untuk mengubah setiap huruf kapital dalam *string* menjadi huruf kecil. Bentuk penulisannya sebagai berikut:

```
strlwr (str);
```

3.3 PERMASALAHAN

(Bermain Dengan Matriks

Karena terlalu fokus dengan hari raya, Holil hampir lupa mengerjakan tugasnya. Holil memiliki tugas materi matriks. Tugas Holil adalah melakukan pencerminan dan rotasi. Pertama, Holil harus melakukan rotasi 90 derajat pada matriks. Setelah itu,dilakukan pencerminan terhadap sumbu X.

```
Masukkan baris: 4
Masukkan kolom: 4
Masukkan elemen matriks:
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
Hasil rotasi 90 derajat searah jarum jam:
13 9 5 1
14 10 6 2
15 11 7 3
16 12 8 4
Hasil pencerminan pada sumbu X:
16 12 8 4
15 11 7 3
14 10 6 2
13 9 5 1
```

```
Masukkan baris: 5
Masukkan kolom: 5
Masukkan elemen matriks:
1 2 3 4 5
2 3 4 5 6
3 4 5 6 7
4 5 6 7 8
5 6 7 8 9
Hasil rotasi 90 derajat searah jarum jam:
5 4 3 2 1
6 5 4 3 2
7 6 5 4 3
8 7 6 5 4
9 8 7 6 5
Hasil pencerminan pada sumbu X:
9 8 7 6 5
8 7 6 5 4
7 6 5 4 3
6 5 4 3 2
5 4 3 2 1
```

Gambar 3.1 Hasil Run Pertama Permasalahan 1

(Enskipsi & Deskripsi)

Setelah mengerjakan tugas, Holil lupa dia belum mengucapkan minal aidin walfaidzin kepada teman-temannya. Namun, Holil ingin ucapan itu ia sampaikan secara rahasia, agar temannya mengeluarkan usaha lebih dalam memahami pesannya. Holil pun menggunakan 2 metode kriptografi yaitu *Reverse cipher*, dan *Caesar cipher*. Holil melakukan enkripsi pada pesannya menggunakan key '5' bergeser ke kanan untuk *Caesar cipher*.

```
Masukkan Pesan: Minal Aidin Walfaidzin
Masukkan Key Caesar Cipher: 5

Hasil Enkripsi dengan Reverse Cipher:
nizdiaflaW nidiA laniM

Hasil enkripsi dengan Reverse Cipher + Caesar Cipher:
sneinfkqfW sninA qfsnM
```

Gambar 3.2 Hasil Run Pertama Permasalahan 2

(Blast The Fireworks)

Holil masih memiliki stok kembang api. Kembang api yang dimiliki Holil memiliki kekuatan ledakan yang berbeda-beda. Ia pun ingin menghabiskan stok kembang apinya. Pertama-tama, Holil akan menentukan jumlah kembang api yang ingin diledakkan, kemudian kembang api akan memiliki *range* ledakan antara 0-9. Setelah itu, Holil menentukan satu kembang api yang akan ia ledakkan terlebih dahulu.

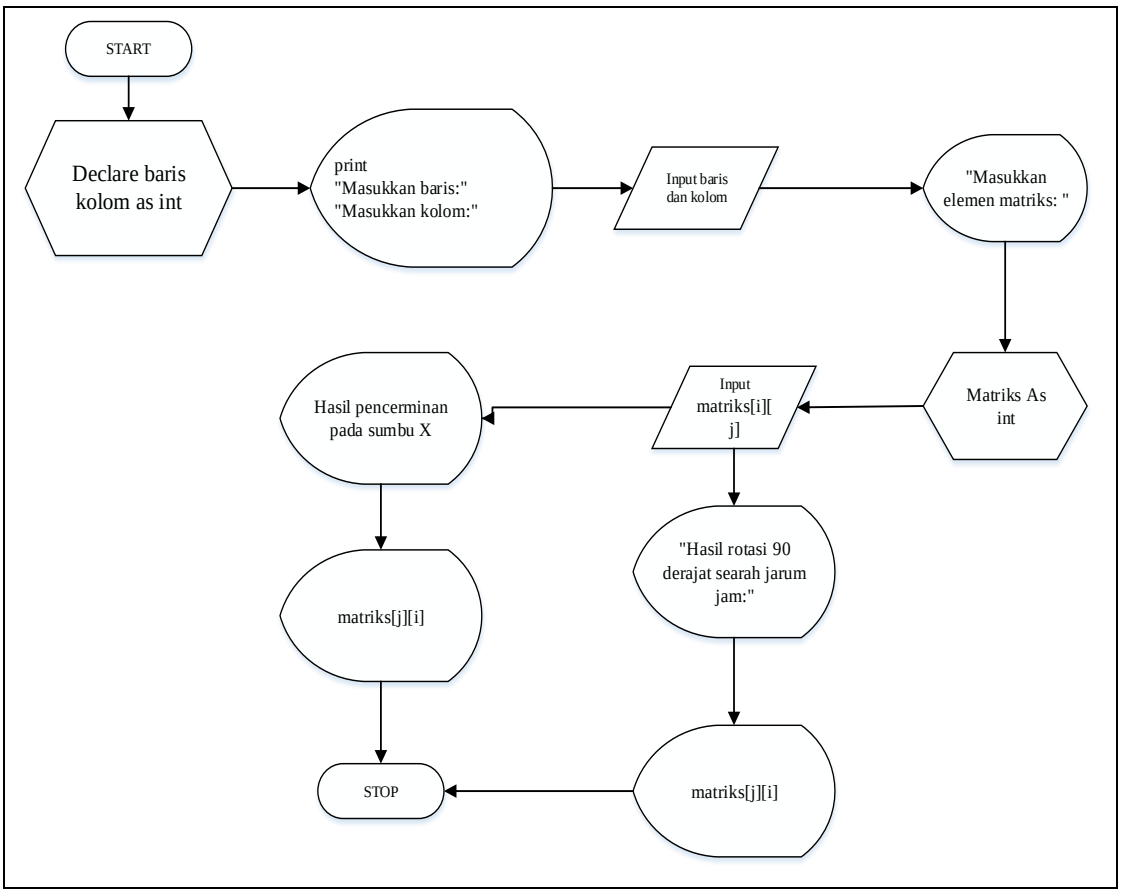
```
Masukkan jumlah bom: 10
 ?  ?  ?  ?  ?  ?  ?  ?  ?  ?
[A][B][C][D][E][F][G][H][I][J]
 1  2  1  0  0  2  1  1  3  0
Masukkan bom yang ingin diledakkan: f
A 1
B 2
C 1
```

Gambar 3.3 Hasil Run Permasalahan 3

3.4 HASIL PERCOBAAN

3.4.1 Program Menghitung Matriks

1. Algoritma
   a. Program dimulai.
   b. Program mendeklarasikan baris dan kolom.
   c. Proram meminta untuk meng*input* jumlah baris dan kolom.
   d. Program mendeklarasikan matriks.
   e. Program meminta *user* meng*input* matriks.
   f. Program Menampilkan hasil matriks setelah diputar 90 derajat searah jarum jam.
   g. Program Menampilkan hasil matriks setelah dilakukan pencerminan pada sumbu X.
2. *Flowchart*



Gambar 3.4 *Flowchart* Program Menghitung Matriks

3. *Source Code*

```
#include <iostream>
using namespace std;

int main(){
int baris, kolom;
    cout << "Masukkan baris: ";
```

```
    cin >> baris;
    cout << "Masukkan kolom: ";
    cin >> kolom;
    cout << "Masukkan elemen matriks: " << endl;
    int matriks[baris][kolom];
    for(int i = 0; i < baris; i++){
        for(int j = 0; j < kolom; j++){
            cin >> matriks[i][j];
        }
    }
    cout << "Hasil rotasi 90 derajat searah jarum jam:" << endl;
    for(int i = 0; i < kolom; i++){
        for(int j = baris - 1; j >= 0; j--){
            cout << matriks[j][i] << " ";
        }
        cout << endl;
    }
    cout << "Hasil pencerminan pada sumbu X:" << endl;
    for(int i = baris - 1; i >= 0; i--){
        for(int j = kolom - 1; j >= 0; j--){
            cout << matriks[j][i] << " ";
        }
        cout << endl;
    }

    return 0;
}
```

4. Hasil Program

```
Masukkan baris: 4
Masukkan kolom: 4
Masukkan elemen matriks:
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
Hasil rotasi 90 derajat searah jarum jam:
13 9 5 1
14 10 6 2
15 11 7 3
16 12 8 4
Hasil pencerminan pada sumbu X:
16 12 8 4
15 11 7 3
14 10 6 2
13 9 5 1
```

Gambar 3.5 Hasil Run Program Menghitung Matriks

Berdasarkan **Gambar 3.5** di atas, dapat diketahui program tersebut menampilkan matriks sesuai dengan *input*an *user*, setelah ditampilkan matriks dilakukan rotasi 90 derajat searah jarum jam dan matriks hasil percerminan pada sumbu x.

3.4.2 Program Mengubah Kalimat

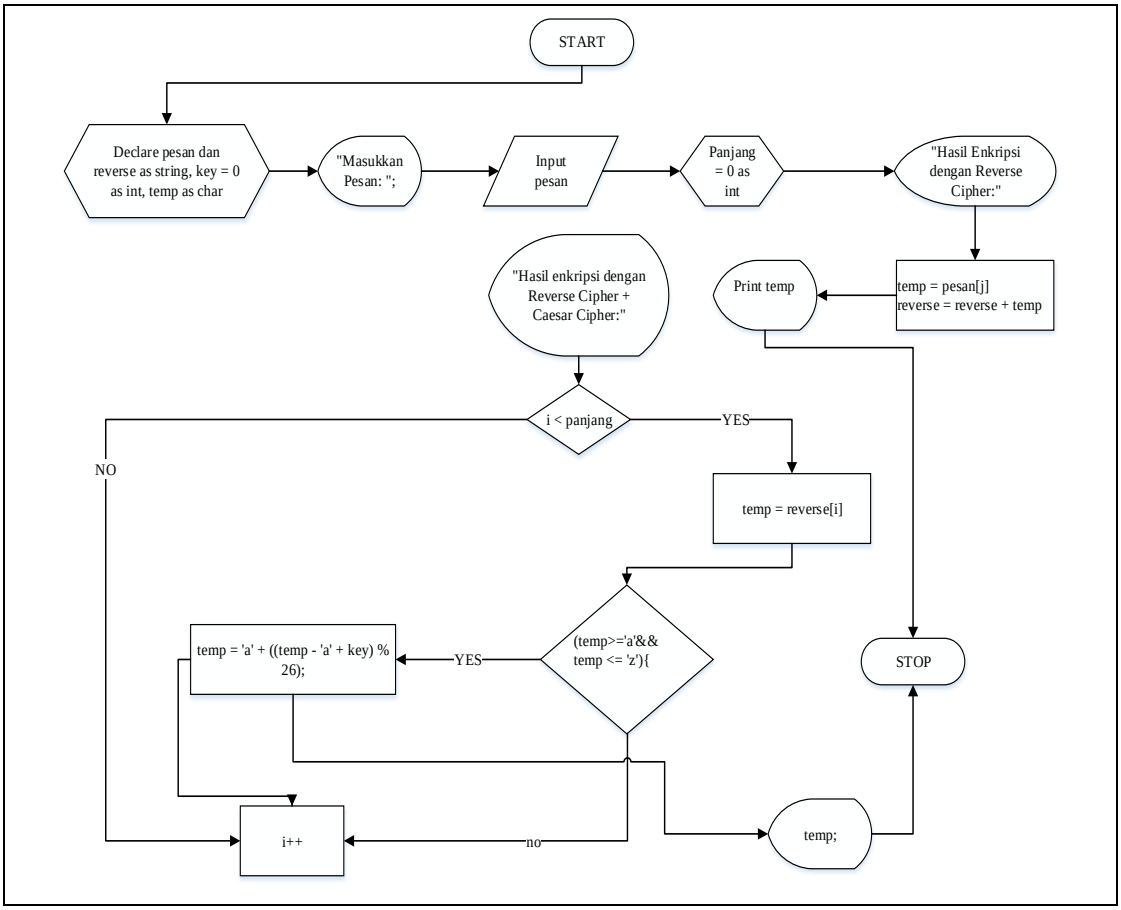
1. Algoritma
   a. Program Dimulai
   b. Program Mendeklarasikan pesan dan *reverse* as *string*, key = 0 as int, temp as char
   c. Program meminta *user* memasukkan *input* pesan
   d. Program mendeklarasikan panjang=0 as int
   e. Program menampilkan key *caesar* chipher

f. Program meminta *user* meg*input* key.

g. Program Menampilkan Hasil enskiripsi dengan *reverse cipher*.

h. Program menampilkan hasil enskripsi dengan *reverse cipher* + *caesar cipher*

2. *Flowchart*



Gambar 3.6 *Flowchart* Program Mengubah Kalimat

3. *Source Code*

```
#include <iostream>
using namespace std;
int main(){
    string pesan, reverse;
    int key = 0;
    char temp;
    cout << "Masukkan Pesan: ";
    getline(cin, pesan);

    int panjang = 0;
    while (pesan[panjang] != '\0') {
        panjang++;
    cout << "Masukkan Key Caesar Cipher: ";
    cin >> key;
     cout << endl;
    cout << "Hasil Enkripsi dengan Reverse Cipher:" << endl;
    for (int j = panjang - 1; j >= 0; j--){
        temp = pesan[j];
        reverse = reverse + temp;
        cout << temp;
```

```
    cout << endl;
    cout << endl << "Hasil enkripsi dengan Reverse Cipher + Caesar
     Cipher:" << endl;
    for (int i = 0; i < panjang; i++){
        temp = reverse[i];
        if (temp >= 'a' && temp <= 'z'){
            temp = 'a' + ((temp - 'a' + key) % 26);
        }
        cout << temp;
}
```

4. Hasil Program

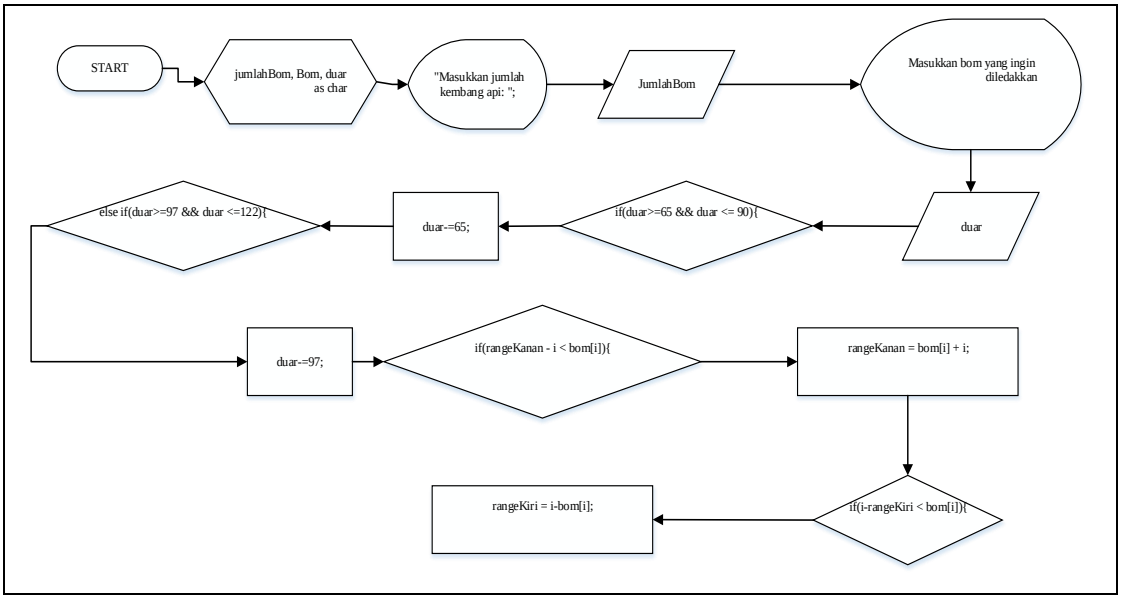
```
Masukkan Pesan: minal aidin walfaidzin
Masukkan Key Caesar Cipher: 5

Hasil Enkripsi dengan Reverse Cipher:
nizdiaflaw nidia lanim

Hasil enkripsi dengan Reverse Cipher + Caesar Cipher:
sneinfkqfb sninf qfsnr
```

Gambar 3.7 Hasil Run Program Mengubah Kalimat

Berdasarkan **Gambar 3.7** di atas, dapat diketahui bahwa program diatas menampilkan *reverse* text yaitu membalikkan kata yang sudah di*input user* lalu menampilkan hasil *reverse* text + *caesar cipher*.

3.4.3 Program Kembang Api

1. Algoritma
   a. Program mendeklarasikan jumlahbom as integer.
   b. Program meminta *user* meng*input* jumlah kembang api.
   c. Program meminta *user* meng*input* jumlah kembang api yang ingin diledakkan.
   d. Program menampilkan sisa kembang api yang tidak terkena ledakkan.
2. *Flowchart*



Gambar 3.8 *Flowchart* Program Kembang Api

3. *Source Code*

```
#include <iostream>
#include <conio.h>
Using namespace std;

int main(){
int jumlahBom;
cout << "Masukkan jumlah kembang api: ";
cin >> jumlahBom;

for(int i='A'; i<'A'+jumlahBom;i++){
if(i=='A'){
for(int j=0;j<jumlahBom;j++){
cout << " ? ";
}
cout << endl;
}
cout << "["<< char(i) << "]";
}
cout << endl;

int bom[jumlahBom];
for(int i=0;i<jumlahBom;i++){
bom[i] = _getch();
bom[i] -= 48;
cout << " " << bom[i] << " ";
}
cout << endl;
char duar;
cout << "Masukkan bom yang ingin diledakkan: ";
cin >> duar;

if(duar>=65 && duar <= 90){
duar-=65;
}
else if(duar>=97 && duar <=122){
duar-=97;
}

//    cout << int(duar);

if(bom[int(duar)]==0){
bom[int(duar)] = -1;
}
else{
int rangeKanan = (int)duar + bom[int(duar)];
int rangeKiri = (int)duar - bom[int(duar)];
int bomYangDiledakkan = bom[int(duar)];

int i = int(duar);
while(i<=rangeKanan){
if(rangeKanan - i < bom[i]){
rangeKanan = bom[i] + i;
}
if(i-rangeKiri < bom[i]){
rangeKiri = i-bom[i];
}

bom[i]= -1;
i++;
```

```
if(rangeKanan> jumlahBom-1){
rangeKanan = jumlahBom-1;
}
}

bom[int(duar)] = bomYangDiledakkan;
i = duar;
while(i>=rangeKiri){
if(i -rangeKiri < bom[i]){
rangeKiri = i - bom[i];
}
bom[i] = -1;
i--;
if(rangeKiri < 0){
rangeKiri = 0;
}
}
}

for(char i='A';i<'A'+jumlahBom;i++){
if(bom[i-65]!=-1){
cout << i << " " << bom[i-65] << endl;
}
}
}}
```

4. Hasil Program

```
Banyak Kembang Api : 10
 ?  ?  ?  ?  ?  ?  ?  ?  ?  ?
[A][B][C][D][E][F][G][H][I][J]
 1  2  1  0  0  2  1  1  3  0
Masukkan bom yang ingin diledakkan: f
Kembang api yang tersisa :
A 1
B 2
C 1
```

Gambar 3.9 Hasil Run Program Kembang Api

Berdasarkan **Gambar 3.9** di atas, dapat diketahui bahwa program diatas menampilkan kembang api beserta jumlah didalamnya, lalu ketika kembang api diledakkan maka akan ditampilkan sisa kembang api-nya.

3.5 ANALISIS DATA

3.5.1 Progam Menghitung Matriks

```
#include<iostream>
```

Script "`#include<iostream>`" adalah *header file* yang digunakan sebagai standar masukan dan keluaran dalam bahasa C++ untuk mendeklarasikan dan mendefinisikan *statement* "`cin`" dan "`cout`".

```
using namespace std;
```

Script "`using namespace std;`" adalah *statement* yang digunakan untuk mendeklarasikan perintah kepada *compiler* bahwa kita akan menggunakan seluruh fungsi yang menjadi bagian dari "`namespace std`".

```
int main ()
```

`int main ()`, Merupakan fungsi (*function*) utama, fungsi yang akan dibaca oleh kompilator pertama kali secara otomatis, tidak akan ada *function* yang otomatis dibaca oleh kompilator kecuali fungsi utama `int main ()` fungsi utama bagaikan sebuah kepala dari sebuah program yang mengatur arah kompilator.

```
int baris, kolom;
```

Script ini mendeklarasikan variabel "`baris, kolom;`" bertipe data "`int`" karena variabel `baris` berisi angka dan sama seperti variabel `kolom` yaitu berisi sekumpulan angka dan huruf.

```
cout << "Masukkan baris: ";
    cin >> baris;
    cout << "Masukkan kolom: ";
    cin >> kolom;
    cout << "Masukkan elemen matriks: " << endl;
```

Script ini terdapat perintah `cout` dipakai untuk menampilkan teks ke layar, yakni salah satu bentuk *output.* Perintah `cout` sendiri merupakan singkatan dari *console out.* Penulisan perintah `cout` harus diikuti karakter panah siku kiri dua kali, yakni karakter <<. Perintah "`cin>>`" yakni meminta data dari *user* / pengguna.

```
int matriks[baris][kolom];
```

Script ini mendeklarasikan variabel "`matriks` bertipe data "`int`" lalu diikuti dengan variabel `[baris][kolom];` yang digunakan untuk mencetak sebuah matriks sesuai dengan *input*an *user* pada *script* sebelumnya

```
for(int i = 0; i < baris; i++){
        for(int j = 0; j < kolom; j++){
            cin >> matriks[i][j];
        }
```

```
    }
```

Script diatas menggunakan perulangan `for` yaitu pernyataan pengulangan yang dikhususkan untuk pengulangan yang secara tepat mengetahui berapa kali pengulangan akan terjadi. Perulangan diatas dilakukan untuk menampilkan hasil matriks dari *input*an *user*

```
    cout << "Hasil rotasi 90 derajat searah jarum jam:" << endl;
    for(int i = 0; i < kolom; i++){
        for(int j = baris - 1; j >= 0; j--){
            cout << matriks[j][i] << " ";
        }
        cout << endl;
```

Script diatas menggunakan perulangan `for` yang diperuntukkan untuk mengubah bentuk matriks yang sebelumnya sudah di-*input* oleh *user* ke bentuk rotasi 90 derajat sesuai permintaan pada bagian `cout` dan bagian `cout << endl` ini digunakan untuk memindahkan kursor ke baris baru *(newline*) dan mem-*flush* (membersihkan) *buffer output.*

```
    cout << "Hasil pencerminan pada sumbu X:" << endl;
    for(int i = baris - 1; i >= 0; i--){
        for(int j = kolom - 1; j >= 0; j--){
            cout << matriks[j][i] << " ";
        }
        cout << endl;
```

Script diatas menggunakan perulangan `for` yang diperuntukkan untuk mengubah matriks awal sesuai dengan *input user* ke hasil perncerminan pada sumbu x sesuai dengan permintaan pada *script* `cout` di kedua perulangan menggunakan pengurangan baris dan kolom masing masing 1.

```
return 0;
```

Syntax return 0; wajib dituliskan dalam setiap kode program mengacu pada struktur bahasa C++ yang benar. Fungsi dari perintah yang satu ini yaitu menutup main() dan mengakhiri program yang sudah ditulis. `Return 0;` sendiri artinya program akan mengembalikan `(return)` nilai 0 ke *operating system* yang menjalankan program tersebut.

3.5.2 Program Mengubah Kalimat

```
#include<iostream>
```

Script "`#include<iostream>`" adalah *header file* yang digunakan sebagai standar masukan dan keluaran dalam bahasa C++ untuk mendeklarasikan dan mendefinisikan *statement* "`cin`" dan "`cout`".

```
using namespace std;
```

Script "`using namespace std;`" adalah *statement* yang digunakan untuk mendeklarasikan perintah kepada *compiler* bahwa kita akan menggunakan seluruh fungsi yang menjadi bagian dari "`namespace std`".

```
int main ()
```

`int main ()`, Merupakan fungsi (*function*) utama, fungsi yang akan dibaca oleh kompilator pertama kali secara otomatis, tidak akan ada *function* yang otomatis dibaca oleh kompilator kecuali fungsi utama `int main ()` fungsi utama bagaikan sebuah kepala dari sebuah program yang mengatur arah kompilator.

```
string pesan, reverse;
    int key = 0;
    char temp;
```

Script ini mendeklarasikan beberapa variabel yang pertama adalah "`pesan, reverse;`" bertipe data "`string`" karena kedua variabel tersebut berisi sekumpulan angka dan huruf. Selanjutnya variabel "`key`" dengan tipe data integer yang hanya bisa meng*input* angka dan terakhir variabel "`temp`" yang berisi huruf.

```
cout << "Masukkan Pesan: ";
    getline(cin, pesan);
```

Pada *Script* ini terdapat perintah *cout* dipakai untuk menampilkan teks ke layar, yakni salah satu bentuk *output.* Perintah *cout* sendiri merupakan singkatan dari *console out.* Penulisan perintah *cout* harus diikuti karakter panah siku kiri dua kali, yakni karakter <<. Perintah "`getline(cin,>>ws,Nama)`" adalah perintah yang mengambil *input* sebanyak satu baris kalimat dan menyimpannya di variabel "`Nama`".

```
    int panjang = 0;
    while (pesan[panjang] != '\0') {
        panjang++;
    }
```

Script diatas menggunakan perulangan *while* adalah perulangan yang akan melakukan perulangan kalau kondisi (syarat) terpenuhi dan sebelumnya dideklarasikan tipe data integer yang sudah di set nilainya dengan 0.

```
cout << "Masukkan Key Caesar Cipher: ";
    cin >> key;

     cout << endl;
```

Script ini terdapat perintah *cout* dipakai untuk menampilkan teks ke layar, yakni salah satu bentuk *output.* Perintah *cout* sendiri merupakan singkatan dari *console out.* Penulisan perintah *cout* harus diikuti karakter panah siku kiri dua kali, yakni karakter <<. Perintah "`cin>>`" yakni meminta data dari *user* / pengguna.

```
    cout << "Hasil Enkripsi dengan Reverse Cipher:" << endl;
    for (int j = panjang - 1; j >= 0; j--){
        temp = pesan[j];
        reverse = reverse + temp;
        cout << temp;
```

```
    }
```

Script diatas menggunakan perulangan `for` untuk mencari `panjang` dari *input* `pesan` pada *script* sebelumnya. *Script* "`temp`" merujuk pada variabel sementara yang digunakan untuk menyimpan nilai sementara selama proses tertentu.

```
cout << endl << "Hasil enkripsi dengan Reverse Cipher + Caesar Cipher:"
     << endl;
```

Script diatas berisi perintah "cout" yang akan menampilkan hasil enskirpsi dari pesan yang sebelumnya sudah di *input* oleh *user* dan hasil *reverse cipher* ditambahkan dengan *caesar cipher* yaitu key yang sudah di *input* juga oleh *user*.

```
    for (int i = 0; i < panjang; i++){
        temp = reverse[i];
        if (temp >= 'a' && temp <= 'z'){
            temp = 'a' + ((temp - 'a' + key) % 26);
        }
        cout << temp;
    }
return 0;
```

Syntax return 0; wajib dituliskan dalam setiap kode program mengacu pada struktur bahasa C++ yang benar. Fungsi dari perintah yang satu ini yaitu menutup main() dan mengakhiri program yang sudah ditulis. `Return 0;` sendiri artinya program akan mengembalikan `(return)` nilai 0 ke operating system yang menjalankan program tersebut.

3.5.3 Program Kembang Api

```
#include<iostream>
```

Script "`#include<iostream>`" adalah *header file* yang digunakan sebagai standar masukan dan keluaran dalam bahasa C++ untuk mendeklarasikan dan mendefinisikan *statement* "`cin`" dan "`cout`".

```
#include <conio.h>
```

Script di atas adalah *header* file program yang mendeklarasikan dan menjalankan perintah I/O konsol dan menampilkan hasil antarmuka. Contoh perintah yang dimiliki oleh *header* ini adalah "`getch()`".

```
using namespace std;
```

Script "`using namespace std;`" adalah *statement* yang digunakan untuk mendeklarasikan perintah kepada *compiler* bahwa kita akan menggunakan seluruh fungsi yang menjadi bagian dari "`namespace std`".

```
int main ()
```

`int main ()`, Merupakan fungsi (*function*) utama, fungsi yang akan dibaca oleh kompilator pertama kali secara otomatis, tidak akan ada *function* yang otomatis dibaca oleh kompilator kecuali fungsi utama `int main ()` fungsi utama bagaikan sebuah kepala dari sebuah program yang mengatur arah kompilator.

```
int jumlahBom;
cout << "Masukkan jumlah bom: ";
cin >> jumlahBom;
```

Script diatas mendeklarasikan variabel "jumlahBom" dengan tipe data integer. *Script* ini terdapat perintah `cout` dipakai untuk menampilkan teks ke layar, yakni salah satu bentuk *output.* Perintah "`cin>>`" yakni meminta data dari *user* / pengguna.

```
for(int i='A'; i<'A'+jumlahBom;i++){
if(i=='A'){
for(int j=0;j<jumlahBom;j++){
cout << " ? ";
}
cout << endl;
}
cout << "["<< char(i) << "]";
}
cout << endl;
```

Script diatas menggunakan perulangan `for` untuk menampilkan jumlah bom sesuai dengan *input*an yang diberikan oleh *user* pada *script* sebelumnya dan dan bagian `cout << endl` ini digunakan untuk memindahkan kursor ke baris baru *(newline)* dan mem-*flush* (membersihkan) *buffer output.*

```
int bom[jumlahBom];
for(int i=0;i<jumlahBom;i++){
bom[i] = _getch();
bom[i] -= 48;
cout << " " << bom[i] << " ";
}
cout << endl;
```

Script di atas mendeklarasikan variabel `bom` yang bertipe data integer yang diikuti dengan perulangan `for` yang digunakan untuk menampilkan "?" sesuai jumlah bom dan menggunakan `getch()` yang berfungsi untuk memungkinkan pembacaan karakter dari *keyboard* tanpa menampilkan karakter tersebut di layar. untuk mengelola *input* pada program berbasis konsol.

```
char duar;
cout << "Masukkan bom yang ingin diledakkan: ";
cin >> duar;
```

Script diatas mendeklarasikan variabel duar dengan tipe data char yaitu tipe data yang hanya dapat menyimpan satu karakter huruf dan terdapat perintah *cout* dipakai untuk

menampilkan teks ke layar dan juga perintah "`cin>>`" yakni meminta data dari *user* / pengguna.

```
if(duar>=65 && duar <= 90){
duar-=65;
}
else if(duar>=97 && duar <=122){
duar-=97;
}
```

Script diatas menggunakan *statement control if else*-if yang berfungsi ketika *user* meng*input* huruf kecil atau huruf besar maka *input* tersebut akan tetap diterima dan terbaca oleh program.

```
if(bom[int(duar)]==0){
bom[int(duar)] = -1;
}
else{
int rangeKanan = (int)duar + bom[int(duar)];
int rangeKiri = (int)duar - bom[int(duar)];
int bomYangDiledakkan = bom[int(duar)];

int i = int(duar);
while(i<=rangeKanan){
if(rangeKanan - i < bom[i]){
rangeKanan = bom[i] + i;
}
if(i-rangeKiri < bom[i]){
rangeKiri = i-bom[i];
}
bom[i]= -1;
i++;
if(rangeKanan> jumlahBom-1){
rangeKanan = jumlahBom-1;
}
```

Script diatas mendeklarasikan variabel `rangeKanan, rangeKiri bomYangDiledakkan` dan `i` dengan tipe data integer, variabel tersebut digunakan untuk mengetahui jumlah kembang api yang meledak di bagian kanan.

```
bom[int(duar)] = bomYangDiledakkan;
i = duar;
while(i>=rangeKiri){
if(i -rangeKiri < bom[i]){
rangeKiri = i - bom[i];
}
bom[i] = -1;
i--;
if(rangeKiri < 0){
rangeKiri = 0;
```

Script diatas menggunakan *statement control if* untuk mengetahui jumlah kembang api yang diledakkan atau terkena radius ledakkan yang berada di bagian kiri hihihi mau tulis apa lagi cobakk

```
for(char i='A';i<'A'+jumlahBom;i++){
if(bom[i-65]!=-1){
cout << i << " " << bom[i-65] << endl;
```

Script diatas menggunakan perulangan for dan *statement control if* yang berfungsi untuk menampilkan bom beserta isi bom yang tidak terkena dampak setelah bom diledakkan

3.6 KESIMPULAN

Berdasarkan hasil praktikum pada Modul 1 tentang Pemrograman Dasar, dapat diambil kesimpulan bahwa:

1. *Array* adalah struktur data yang menyimpan sekumpulan elemen yang bertipe sama. Setiap elemen dapat diakses secara langsung melalui indeksnya. Indeks dari *array* haruslah bertipe data yang memiliki *successor* dan *predecessor* seperti integer atau *character*.
2. Nilai dalam *Array* dapat digunakan dalam sebuah program dengan cara memanggil *array* menggunakan indeksnya. Indeks dari *array* dimulai dari 0 hingga ukuran *array* dikurangi satu.
3. Terdapat beberapa fungsi manipulasi *string* yang dapat digunakan, seperti “strlwr”, “strupr”, “strlen”, dan lain-lain. “strlwr” digunakan untuk mengubah seluruh karakter menjadi huruf kecil, “strupr” digunakan untuk mengubah seluruh karakter menjadi huruf kapital, dan “strlen” digunakan untuk mencari panjang *string*.

DAFTAR PUSTAKA

Bachtiar, Adam Mukhairil (2017) Pemrograman C & C++. Bandung: Politeknik Telkom Bandung.

Harumy, T. Henny Febriana, dkk. (2016). Belajar Dasar Algoritma & Pemrograman C++. Yogyakarta: Deepublish.

Suarga. (2012). Algoritma dan Pemrograman. Yogyakarta: CV ANDI OFFSET.

Warnilah, A. I. (2015). Modul Algoritma Dan Pemrograman. Akademi Manajemen Informatika & Komputer (AMIK – BSI).