

LAPORAN PRAKTIKUM

Matakuliah	Struktur Data
Pertemuan ke	5
Nama Praktikan	Wijayanto Agung Wibowo
NIM	22.111.4552
NILAI (diisi oleh dosen / asisten praktikum)	

A. Tujuan

Setelah praktikum ini, praktikan diharapkan dapat:

1. Memahami tipe data dasar
2. Memahami tipe data bentukan
3. Stuktur Program menggunakan bahasa C++ Praktikum

B. Hasil Percobaan

1. Percobaan 1

a) Tampilan Coding

```
1  #include <iostream>
2  using namespace std;
3
4  int maksimal = 5;
5  string arrayBuku[5];
6  int top = 0;
7
8  bool isFull() {
9      if (top == maksimal) {
10         return true;
11     }
12     else {
13         return false;
14     }
15 }
16
17 bool isEmpty() {
18     if (top == 0) {
19         return true;
20     }
21     else {
22         return false;
23     }
24 }
25
26 void pushArray(string data) {
27     if (isFull()) {
28         cout << "Data penuh" << endl;
29     }
30     else {
31         arrayBuku[top] = data;
32         top++;
33     }
34 }
35
36 void popArray() {
37     if (isEmpty()) {
38         cout << "Data kosong!" << endl;
39     }
40     else {
41         arrayBuku[top - 1] = "";
42         top--;
43     }
44 }
```

```

44     }
45
46     void displayArray() {
47         if (isEmpty()) {
48             cout << "Data kosong!!" << endl;
49         }
50         else {
51             cout << "Data stack array:" << endl;
52             cout << "-----" << endl;
53             for (int i = maksimal - 1; i >= 0; i--) {
54                 if (arrayBuku[i] != "") {
55                     cout << "Stack " << i << ": " << arrayBuku[i] << endl;
56                 }
57             }
58             cout << "\n" << endl;
59         }
60     }
61
62     void destroyArray() {
63         for (int i = 0; i < top; i++) {
64             arrayBuku[i] = "";
65         }
66         top = 0;
67     }
68
69     int main() {
70         pushArray("Delapan");
71         displayArray();
72
73         pushArray("Sembilan");
74         pushArray("Tiga");
75         displayArray();
76
77         popArray();
78         displayArray();
79
80         pushArray("Lima");
81         pushArray("Enam");
82         pushArray("Empat");
83         pushArray("Tujuh");
84         displayArray();
85
86         popArray();
87         displayArray();
88
89         cout << "Apakah data full? :" << isFull() << endl;
90         cout << "Apakah data kosong? :" << isEmpty() << endl << endl;
91
92         destroyArray();
93
94         cout << "Setelah di clear " << endl;
95         cout << "Apakah data full? : " << isFull() << endl;
96         cout << "Apakah data kosong? : " << isEmpty() << endl;
97         system("pause");
98         return 0;
99     }

```

b) Hasil Running

```
NAMA : WIJAYANTO AGUNG WIBOWO
NIM : 22.11.4552

Data stack array:
-----
Stack 0: Delapan

Data stack array:
-----
Stack 2: Tiga
Stack 1: Sembilan
Stack 0: Delapan

Data stack array:
-----
Stack 1: Sembilan
Stack 0: Delapan

Data penuh
Data stack array:
-----
Stack 4: Empat
Stack 3: Enam
Stack 2: Lima
Stack 1: Sembilan
Stack 0: Delapan

Data stack array:
-----
Stack 3: Enam
Stack 2: Lima
Stack 1: Sembilan
Stack 0: Delapan

Apakah data full? :0
Apakah data kosong? :0

Setelah di clear
Apakah data full? : 0
Apakah data kosong? : 1
Press any key to continue . . .
```

c) Penjelasan

Penjelasan fungsinya yaitu:

- a. `isFull` : Untuk mengecek full atau belum isi stacknya. Dengan mengecek top dari stack dan nilai maksimal stack. Apabila sama maka fungsi `isFull` akan memberikan respon `true`.
- b. `isEmpty` : Untuk mengecek kosong tidaknya isi stack. Dengan mengecek top dari stack dan nilai bottom dari stack yaitu 0. Apabila sama maka fungsi `isEmpty` akan memberikan respon `true`.
- c. `pushArray`: untuk memasukan data ke stack. Apabila stack belum full, dan menambah nilai top ke +1.
- d. `popArray`: untuk mengeluarkan data dari stack dan mengurangi nilai top ke -1. Apabila data sudah kosong maka akan ada keterangan data kosong.
- e. `displayArray`: untuk menunjukan isi dari stack array. Apabila data kosong, maka akan ada keterangan data kosong.
- f. `destroyArray`: untuk mengosongkan data dari stack dan akan membuat top ke index ke 0;

Di dalam praktikum tadi, kita mengpush data ke stack dengan fungsi `pushArray(">data<")`, dan mengkombinasikan fungsi dari penjelasan diatas kedalam implementasi koding. Dimana kita bisa mengecek, mempush data ke array stack, mengecek kondisi stack dan mendisplay isi dari stack.

2. Percobaan 2

a) Tampilan Coding

```
1  #include <iostream>
2  using namespace std;
3
4  const int MAX_STACK = 5;
5
6  struct Stack
7  {
8      int top;
9      int data[MAX_STACK];
10 };
11
12 //prototipe fungsi stack
13 void inisialisasi();
14 void push(int data); //menambahkan item pada stack
15 void pop(); //menghapus item pada stack
16 void clear(); //mengosongkan stack
17 bool isEmpty(); //untuk mengecek apakah stack kosong
18 bool isFull(); //untuk mengecek apakah stack penuh
19 void print(); //mencetak item stack
20
21 Stack stack; //deklarasi var stack dg tipe struck Stack
22
23
24 int main() {
25     inisialisasi();
26
27     int pilihanMenu;
28     int data;
29
30     do
31     {
32         cout << "NAMA : WIJAYANTO AGUNG WIBOWO" <<endl;
33         cout << " NIM : 22.11.4552" <<endl <<endl;
34         cout <<">>> PILIHAN MENU STACK <<<" <<endl <<endl;
35         cout << "1. Menambah item stack" <<endl;
36         cout << "2. Menghapus item stack" <<endl;
37         cout << "3. Menampilkan item stack" <<endl;
38         cout << "4. Mengosongkan stack" <<endl;
39         cout << "5. Selesai" <<endl <<endl;
40
41         cout << "Masukan pilihan Anda : "; cin >> pilihanMenu;
42         cout << endl;
43
44         switch (pilihanMenu)
45         {
46             case 1:
47                 cout << "Masukan data : "; cin >> data;
48                 push(data);
49                 break;
50
51             case 2:
52                 pop();
53                 break;
54
55             case 3:
56                 print();
57                 break;
```

```

58
59         case 4:
60             clear();
61             break;
62     }
63     } while (pilihanMenu != 5);
64
65     cout << endl;
66
67     system("pause");
68     return 0;
69
70 }
71
72 void inisialisasi()
73 {
74     stack.top = -1;
75 }
76
77 void push(int data)
78 {
79     stack.top++;
80     stack.data[stack.top] = data;
81     cout << "Data berhasil ditambahkan" <<endl <<endl;
82 }
83
84 void pop()
85 {
86     cout <<"Data " <<stack.data[stack.top] << " sudah dihapus" <<endl <<endl;
87     stack.top--;
88 }
89
90 void clear()
91 {
92     stack.top = -1;
93     cout << "Stack sudah dikosongkan" <<endl <<endl;
94 }
95
96 bool isEmpty()
97 {
98     return (stack.top == -1);
99 }
100
101 bool isFull()
102 {
103     return (stack.top >= (MAX_STACK -1));
104 }
105
106 void print()
107 {
108     cout << "Isi stack :" <<endl <<endl;
109     for (int i = stack.top; i >= 0; i--)
110     {
111         cout <<stack.data[i] <<endl;
112     }
113     cout <<endl <<endl;
114 }

```

b) Hasil Running

```
Masukan pilihan Anda : 1
Masukan data : 7
Data berhasil ditambahkan

NAMA : WIJAYANTO AGUNG WIBOWO
NIM : 22.11.4552

>>> PILIHAN MENU STACK <<<

1. Menambah item stack
2. Menghapus item stack
3. Menampilkan item stack
4. Mengosongkan stack
5. Selesai

Masukan pilihan Anda : 3

Isi stack :

7
6
5
4
3
2
1
```

c) Penjelasan

Pilihan eksekusi proses diatas, bila memilih menambah item stack, maka inputan data kita akan dimasukkan kedalam stack. Bila memilih menghapus item stack, maka data <top> akan dihapus dan top akan -1. Bila memilih menampilkan item stack, maka akan menampilkan Semua stack sampai ke posisi top. Bila memilih mengosongkan stack, maka akan mengosongkan Semua stack dan posisi top akan kembali ke 0.

3. Studi Kasus

a) Tampilan coding

```
1  #include <iostream>
2  #include <string>
3  #include <vector>
4  using namespace std;
5  class Antrian
6  {
7      private:
8          vector<string> data;
9          int depan, belakang;
10         int maksElemen;
11
12     public:
13         // Konstruktor
14         Antrian(int ukuran)
15         {
16             depan = 0;
17             belakang = 0;
18             maksElemen = ukuran;
19             data.resize(ukuran); // Ukuran vector
20         }
21
22         // Memasukkan data ke antrian
23         // Nilai balik tidak ada
24         void insert(string x)
25         {
26             int posisiBelakang;
27             // Geser belakang ke posisi berikutnya
28             if (belakang == maksElemen)
29                 posisiBelakang = 1;
30             else
31                 posisiBelakang = belakang + 1;
32             // Cek belakang apa sama dengan Depan
33             if (posisiBelakang == depan)
34                 cout << "Antrian penuh" << endl;
35             else
36             {
37                 belakang = posisiBelakang;
38                 // Masukkan data
39                 data[belakang] = x;
40             }
41         }
42
43         string remove(void)
44         {
45             if (empty())
46             {
47                 cout << "Antrian kosong" << endl;
48                 return "";
49             }
50             if (depan == maksElemen)
51                 depan = 1;
52             else
53                 depan = depan + 1;
54             return data[depan];
55         }
56     }
```

```

56
57     bool empty(void)
58     {
59         if (depan == belakang)
60             return true;
61         else
62             return false;
63     }
64 };
65
66 int main()
67 {
68     cout << "NAMA : WIJAYANTO AGUNG WIBOWO" <<endl;
69     cout << " NIM : 22.11.4552" <<endl <<endl;
70     int ukuran = 10;
71
72     Antrian daftar(ukuran); // Buat objek
73     // Masukkan 5 buah nama
74     daftar.insert("Aman");
75     daftar.insert("Budi");
76     daftar.insert("Caca");
77     daftar.insert("Didi");
78     daftar.insert("Edi");
79     // Kosongkan isi antrian dan tampilkan
80     while (!daftar.empty())
81     {
82         string nama = daftar.remove();
83         cout << nama << endl;
84     }
85     return 0;
86 }
87

```

b) Hasil running

```

NAMA : WIJAYANTO AGUNG WIBOWO
NIM : 22.11.4552

Aman
Budi
Caca
Didi
Edi

-----
Process exited after 0.06394 seconds with return value 0
Press any key to continue . . .

```

c) Penjelasan

Memasukan `daftar.insert(">data<")` kedalam struct. Dan menampilkan Semua data jika kondisi `daftar.empty()` false atau masih ada data tersedia.

4. Latihan 1

a) Tampilan coding

```
1  #include <iostream>
2  using namespace std;
3
4  int maksimal = 5;
5  string arrayBuku[5];
6  int top = 0;
7
8  bool isFull() {
9      if (top == maksimal) {
10         return true;
11     }
12     else {
13         return false;
14     }
15 }
16
17 bool isEmpty() {
18     if (top == 0) {
19         return true;
20     }
21     else {
22         return false;
23     }
24 }
25
26 void pushArray(string data) {
27     if (isFull()) {
28         cout << "Data penuh" << endl;
29     }
30     else {
31         arrayBuku[top] = data;
32         top++;
33     }
34 }
35
36 void popArray() {
37     if (isEmpty()) {
38         cout << "Data kosong!" << endl;
39     }
40     else {
41         arrayBuku[top - 1] = "";
42         top--;
43     }
44 }
```

```

45 void displayArray() {
46     if (isEmpty()) {
47         cout << "Data kosong!!" << endl;
48     }
49     else {
50         cout << "Data stack array:" << endl;
51         cout << "-----" << endl;
52         for (int i = maksimal - 1; i >= 0; i--) {
53             if (arrayBuku[i] != "") {
54                 cout << "Stack " << i << ": " << arrayBuku[i] << endl;
55             }
56         }
57         cout << "\n" << endl;
58     }
59 }
60
61 void destroyArray() {
62     for (int i = 0; i < top; i++) {
63         arrayBuku[i] = "";
64     }
65     top = 0;
66 }
67
68
69 int main() {
70     int pilihanMenu;
71     string data;
72     cout << "NAMA : WIJAYANTO AGUNG WIBOWO" << endl;
73     cout << "NIM : 22.11.4552" << endl << endl;
74     do
75     {
76
77         cout << ">>> PILIHAN MENU STACK <<<" << endl << endl;
78         cout << "1. Menambah item stack" << endl;
79         cout << "2. Menghapus item stack" << endl;
80         cout << "3. Menampilkan item stack" << endl;
81         cout << "4. Mengosongkan stack" << endl;
82         cout << "5. Selesai" << endl << endl;
83
84         cout << "Masukan pilihan Anda : "; cin >> pilihanMenu;
85         cout << endl;
86         switch (pilihanMenu)
87         {
88             case 1:
89                 if (isFull()){
90                     cout << "DATA SUDAH PENUH !!!!!!!!!!" << endl;
91                 } else{
92                     cout << "Masukan data : "; cin >> data;
93                     pushArray(data);
94                     break;
95                 }
96
97             case 2:
98                 popArray();
99                 break;
100
101             case 3:
102                 displayArray();
103                 break;
104
105             case 4:
106                 destroyArray();
107                 break;
108         }
109     } while (pilihanMenu != 5);
110
111     cout << endl;
112     system("pause");
113     return 0;
114     system("pause");
115     return 0;
116 }
117
118

```

b) Hasil running

<pre> Masukan pilihan Anda : 1 Masukan data : 4 NAMA : WIJAYANTO AGUNG WIBOWO NIM : 22.11.4552 >>> PILIHAN MENU STACK <<< 1. Menambah item stack 2. Menghapus item stack 3. Menampilkan item stack 4. Mengosongkan stack 5. Selesai Masukan pilihan Anda : 1 Masukan data : 5 NAMA : WIJAYANTO AGUNG WIBOWO NIM : 22.11.4552 >>> PILIHAN MENU STACK <<< 1. Menambah item stack 2. Menghapus item stack 3. Menampilkan item stack 4. Mengosongkan stack 5. Selesai Masukan pilihan Anda : 1 DATA SUDAH PENUH !!!!!!!!!!! NAMA : WIJAYANTO AGUNG WIBOWO NIM : 22.11.4552 >>> PILIHAN MENU STACK <<< 1. Menambah item stack 2. Menghapus item stack 3. Menampilkan item stack 4. Mengosongkan stack 5. Selesai Masukan pilihan Anda : _ </pre>	<pre> NAMA : WIJAYANTO AGUNG WIBOWO NIM : 22.11.4552 >>> PILIHAN MENU STACK <<< 1. Menambah item stack 2. Menghapus item stack 3. Menampilkan item stack 4. Mengosongkan stack 5. Selesai Masukan pilihan Anda : 3 Data kosong!! >>> PILIHAN MENU STACK <<< 1. Menambah item stack 2. Menghapus item stack 3. Menampilkan item stack 4. Mengosongkan stack 5. Selesai Masukan pilihan Anda : </pre>
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

c) Penjelasan

Memilih angka 1 akan membuat proses dari stack akan berfungsi dan akan membuat inputan dari user masuk kedalam stack, apabila stack penuh(dimana top nya adalah 5) maka akan muncul peringatan data sudah penuh. Apabila memilih angka 3, dan jika tidak ada data, maka akan memunculkan peringatan "Data kosong" dimana posisi top ada di nilai 0.

5. Latihan 2

1. Ilustrasikan antrian untuk daftar pemanggilan operasi berikut:
store('Ana')
store('Bina')
store('Cika')
retrieve store('Dana')
retrieve store('Eka')
store('Fina')
retrieve

→

Operasi	Elemen yang disimpan	Head	Tail
Store	'Ana'	0	0
Store	'Bina'	0	1
Store	'Cika'	0	2
Retrieve	'Ana'	1	2
Store	'Dana'	1	3
Retrieve	'Bina'	2	3
Store	'Eka'	2	4
Store	'Fina'	2	5
Retrieve	'Cika'	3	5

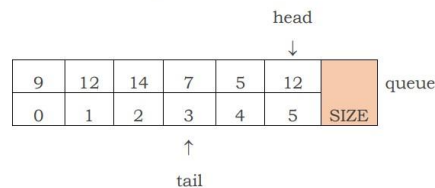
2. Saat kita mengalokasikan antrian menggunakan array dengan jumlah n elemen maka data yang dapat kita isikan hanya $n-1$ saja, mengapa?

→ Saat kita mengalokasikan antrian menggunakan array dengan jumlah n elemen, data yang dapat kita isikan hanya $n-1$ saja karena kita perlu menyisakan satu elemen di array untuk menandai posisi awal dan akhir dari antrian.

Dalam implementasi antrian menggunakan array, posisi awal antrian biasanya diwakili oleh variabel "front", sedangkan posisi akhir antrian diwakili oleh variabel "rear". Saat antrian kosong, kedua variabel ini diatur ke nilai -1. Ketika kita menambahkan elemen ke antrian, kita akan menambahkan elemen tersebut ke posisi $\text{rear} + 1$ dan kemudian mengubah nilai rear menjadi $\text{rear} + 1$. Saat kita mengambil elemen dari antrian, kita akan mengambil elemen pada posisi front dan kemudian mengubah nilai front menjadi $\text{front} + 1$.

Dalam implementasi ini, kita perlu menyisakan satu elemen di array untuk menandai bahwa antrian kosong (yaitu ketika front dan rear sama-sama bernilai -1) atau penuh (yaitu ketika rear berada di elemen terakhir di array). Oleh karena itu, jika kita mengalokasikan array dengan n elemen, maka data yang dapat kita isikan hanyalah $n-1$ elemen saja.

3. Terdapat ilustrasi antrian sebagai berikut :



Sebutkan jumlah dari data antrian diatas dan bagaimana urutan antriannya?
Apabila ditambahkan operasi remove diakhir proses maka gambarkan ilustrasi hasil akhirnya.

Jumlah data antrian di atas adalah 6. Urutan antriannya adalah: 9, 12, 14, 7, 5, 12.

Apabila ditambahkan operasi remove di akhir proses, maka antrian akan menjadi:

12	14	7	5	12	SIZE
1	2	3	4	5	


4. Apakah memungkinkan dilakukan penambahan elemen didepan?
Mengapa?

→ Penambahan elemen di depan antrian dikenal sebagai operasi "enqueue-front". Tidak semua struktur data antrian mendukung operasi enqueue-front secara langsung. Struktur data antrian dasar seperti array

atau list sederhana tidak mendukung operasi enqueue-front karena menyebabkan pergeseran data di dalam antrian yang sudah ada.

Namun, ada beberapa struktur data antrian yang mendukung operasi enqueue-front, seperti deque (double-ended queue). Deque mendukung penambahan elemen di awal dan di akhir antrian secara efisien karena menggunakan struktur data linked list ganda. Namun, penambahan elemen di depan antrian dapat menyebabkan perubahan urutan dan posisi data dalam antrian, sehingga perlu dipertimbangkan secara matang dalam merancang aplikasi.

5. Modifikasi source code antrian diatas supaya dapat melakukan penambahan dan penghapusan elemen saat program berjalan.



```
*D:\Strukdat\STACK & QUEUE\sourcecodeantrian.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Ma
binary.cpp x sort.cpp x new 2.cpp x oo.cpp x hai.cpp x
1 #include <iostream>
2 #include <string>
3 #include <vector>
4 using namespace std;
5
6 class Antrian {
7 private:
8     vector<string> data;
9     int depan, belakang;
10    int maksElemen;
11 public:
12     // Konstruktor
13     Antrian(int ukuran) {
14         depan = 0;
15         belakang = 0;
16         maksElemen = ukuran;
17         data.resize(ukuran); // Ukuran vector
18     }
19
20     // Memasukkan data ke antrian
21     void insert(string x) {
22         int posisiBelakang;
23         // Geser belakang ke posisi berikutnya
24         if (belakang == maksElemen)
25             posisiBelakang = 1;
26         else
27             posisiBelakang = belakang + 1;
28         // Cek belakang apa sama dengan Depan
29         if (posisiBelakang == depan)
30             cout << "Antrian penuh" << endl;
31         else {
32             belakang = posisiBelakang;
33             // Masukkan data
34             data[belakang] = x;
35         }
36     }
37
38     // Menghapus data dari antrian
39     string remove() {
40         if (empty()) {
41             cout << "Antrian kosong" << endl;
42             return "";
43         }
44         if (depan == maksElemen)
```

```

43         }
44         if (depan == maksElemen)
45             depan = 1;
46         else
47             depan = depan + 1;
48         return data[depan];
49     }
50
51     // Cek apakah antrian kosong
52     bool empty() {
53         if (depan == belakang)
54             return true;
55         else
56             return false;
57     }
58 };
59
60 int main() {
61     int ukuran = 10;
62     Antrian daftar(ukuran); // Buat objek
63     string nama;
64     int pilihan;
65
66     do {
67         // Tampilkan menu
68         cout << "Menu Antrian" << endl;
69         cout << "1. Masukkan data" << endl;
70         cout << "2. Keluarkan data" << endl;
71         cout << "3. Keluar" << endl;
72         cout << "Pilihan: ";
73         cin >> pilihan;
74
75         switch(pilihan) {
76             case 1: // Masukkan data
77                 cout << "Masukkan nama: ";
78                 cin >> nama;
79                 daftar.insert(nama);
80                 break;
81             case 2: // Keluarkan data
82                 nama = daftar.remove();
83                 if (nama != "")
84                     cout << "Data yang dikeluarkan: " << nama << endl;
85                 break;
86             case 3: // Keluar
87
88                 else
89                     return false;
90             }
91         }
92     };
93
94     int main() {
95         int ukuran = 10;
96         Antrian daftar(ukuran); // Buat objek
97         string nama;
98         int pilihan;
99
100        do {
101            // Tampilkan menu
102            cout << "Menu Antrian" << endl;
103            cout << "1. Masukkan data" << endl;
104            cout << "2. Keluarkan data" << endl;
105            cout << "3. Keluar" << endl;
106            cout << "Pilihan: ";
107            cin >> pilihan;
108
109            switch(pilihan) {
110                case 1: // Masukkan data
111                    cout << "Masukkan nama: ";
112                    cin >> nama;
113                    daftar.insert(nama);
114                    break;
115                case 2: // Keluarkan data
116                    nama = daftar.remove();
117                    if (nama != "")
118                        cout << "Data yang dikeluarkan: " << nama << endl;
119                    break;
120                case 3: // Keluar
121                    cout << "Keluar dari program" << endl;
122                    break;
123                default:
124                    cout << "Pilihan tidak valid" << endl;
125                    break;
126            }
127        } while (pilihan != 3);
128
129        return 0;
130    }
131 }

```

```

D:\Strukdat>cd "STACK & QUEUE"

D:\Strukdat\STACK & QUEUE>g++ sourcecodeantrian.c

D:\Strukdat\STACK & QUEUE>sourcecodeantrian.exe
Menu Antrian
1. Masukkan data
2. Keluarkan data
3. Keluar
Pilihan: 1
Masukkan nama: windy
Menu Antrian
1. Masukkan data
2. Keluarkan data
3. Keluar
Pilihan: 1
Masukkan nama: vina
Menu Antrian
1. Masukkan data
2. Keluarkan data
3. Keluar
Pilihan: 1
Masukkan nama: dini
Menu Antrian
1. Masukkan data
2. Keluarkan data
3. Keluar
Pilihan: 1
Masukkan nama: sheila
Menu Antrian
1. Masukkan data
2. Keluarkan data
3. Keluar
Pilihan: 1
Masukkan nama: alvin

```

```
3. Keluar
Pilihan: 1
Masukkan nama: alvin
Menu Antrian
1. Masukkan data
2. Keluarkan data
3. Keluar
Pilihan: 2
Data yang dikeluarkan: windy
Menu Antrian
1. Masukkan data
2. Keluarkan data
3. Keluar
Pilihan: 2
Data yang dikeluarkan: vina
Menu Antrian
1. Masukkan data
2. Keluarkan data
3. Keluar
Pilihan: 2
Data yang dikeluarkan: dini
Menu Antrian
1. Masukkan data
2. Keluarkan data
3. Keluar
Pilihan: 2
Data yang dikeluarkan: sheila
Menu Antrian
1. Masukkan data
2. Keluarkan data
3. Keluar
Pilihan: 2
Data yang dikeluarkan: alvin
Menu Antrian
1. Masukkan data
2. Keluarkan data
3. Keluar
Pilihan: 2
```

```
Pilihan: 2
Antrian kosong
Menu Antrian
1. Masukkan data
2. Keluarkan data
3. Keluar
Pilihan: 3
Keluar dari program

D:\Strukdat\STACK & QUEUE>
```

C. Kesimpulan

Setelah melakukan percobaan pada Latihan 1 sampai Latihan 2 saya dapat memahami bahwa stack dan queue itu hampir sama. Perbedaannya terletak pada pengambilan datanya, dimana queue mengambil data di head sedangkan stack itu mengambil data melewati top.

Kesamaan dari queue dan stack terletak pada pemanfaatan array dan mengalokasikan alamat memory itu mekanismenya sama.