

KODE UTP PRAKTIKUM ASD

NAMA:KEVIN BETESDA KORNELIUS BANGUN

NIM:1203230019

KELAS:IF 03-02

1.QUEUE

2.QUEUE

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAX_SIZE 10
5
6 typedef struct Queue {
7     int data[MAX_SIZE];
8     int front;
9     int rear;
10 } Queue;
11
12 void initQueue(Queue *queue) {
13     queue->front = -1;
14     queue->rear = -1;
15 }
16
17 int isEmpty(Queue *queue) {
18     return (queue->front == -1 && queue->rear == -1);
19 }
20
21 int isFull(Queue *queue) {
22     return ((queue->rear + 1) % MAX_SIZE == queue->front);
23 }
24
25 void enqueue(Queue *queue, int value) {
26     if (isFull(queue)) {
27         printf("queue is full\n");
28         return;
29     }
30     if (isEmpty(queue)) {
31         queue->front = queue->rear = 0;
32     } else {
33         queue->rear = (queue->rear + 1) % MAX_SIZE;
34     }
35 }
```

```
26 void enqueue(Queue *queue, int value) {
27     if (isFull(queue)) {
28         printf("queue is full\n");
29         return;
30     }
31     queue->rear = (queue->rear + 1) % MAX_SIZE;
32     queue->data[queue->rear] = value;
33 }
34
35 int dequeue(Queue *queue) {
36     if (isEmpty(queue)) {
37         printf("queue is empty\n");
38         return -1;
39     }
40     int value = queue->data[queue->front];
41     if (queue->front == queue->rear) {
42         queue->front = queue->rear = -1;
43     } else {
44         queue->front = (queue->front + 1) % MAX_SIZE;
45     }
46     return value;
47 }
48
49 void handlePatient(Queue *queue, int severity) {
50     if (severity <= 5) {
51         printf("ditangani dokter umum\n");
52     } else {
53         printf("ditangani dokter spesialis\n");
54     }
55 }
56
57 int main() {
58     Queue queue;
59     initQueue(&queue);
60 }
```

```
File Edit Selection View Go Run Terminal Help
ALGORITHM STRUKTUR DATA

EXPLORER
SEARCH
ALGORITHM...
.vscode
ASD.c
  asdweek5.c
  asdweek7.c
  OTHCIRCULAR.c
  OTSSTACK.c
  OTSSTACK.exe
  otsweek7-queue.c
  otsweek7-queue.exe
  rekursif.c
  rekursif.exe
  struct.c
  struct.exe
  struct2.c
  struct2.exe
  tempCodeRunnerFile.c
  TUGAS3.c
  week5ots.c
  week5ots.exe
  ASD2.c
  ASDTEORI.c
  ASDujn
    ASDPRTKM.zip
    queue.c
    queue.exe
    queue2.c
    queue2.exe
    stack.c
    STACKELEMEN.c
  OUTLINE
  TIMELINE

ASDujn > C queue2c > ...
41 int dequeue(Queue *queue) {
55     return value;
56 }
57
58 void handlePatient(Queue *queue, int severity) {
59     if (severity <= 5) {
60         printf("ditangani dokter umum\n");
61     } else {
62         printf("ditangani dokter spesialis\n");
63     }
64 }
65
66 int main() {
67     Queue queue;
68     initQueue(&queue);
69
70     int n, i, severity;
71     printf("Input jumlah pasien: ");
72     scanf("%d", &n);
73
74     printf("Input tingkat keparahan pasien: \n");
75     for (i = 0; i < n; i++) {
76         scanf("%d", &severity);
77         enqueue(&queue, severity);
78     }
79
80     printf("Output:\n");
81     for (i = 0; i < n; i++) {
82         severity = dequeue(&queue);
83         handlePatient(&queue, severity);
84     }
85
86     return 0;
87 }
88
```

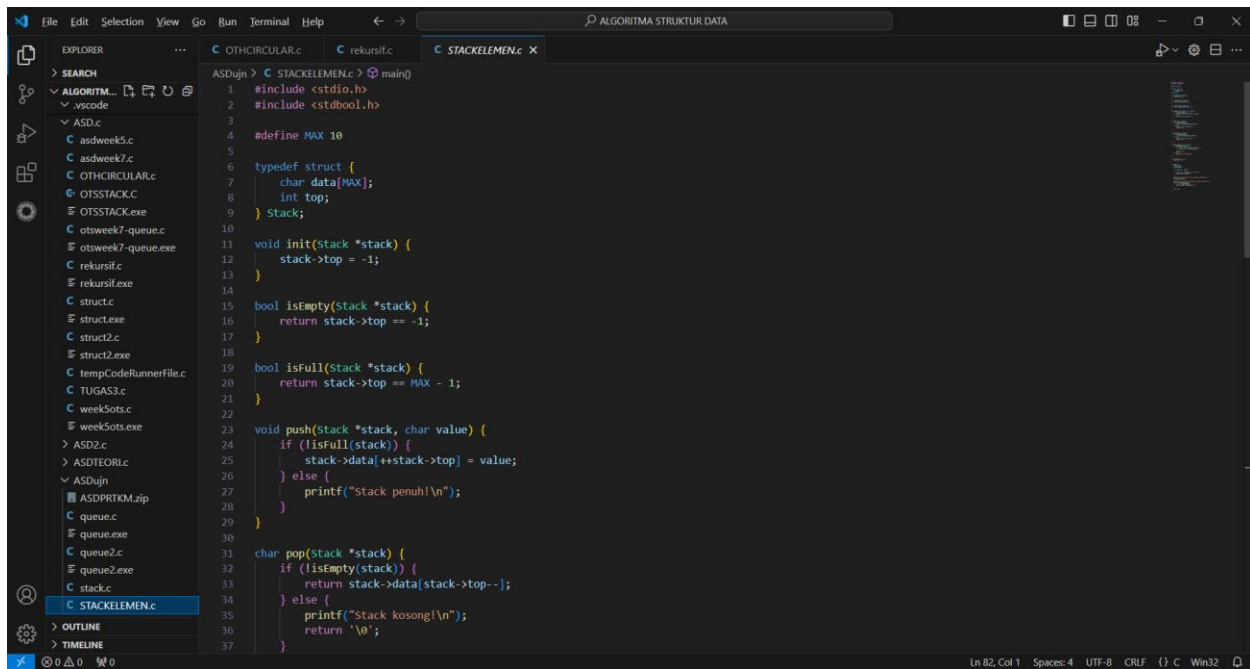
3.STACK

```
File Edit Selection View Go Run Terminal Help
ALGORITHM STRUKTUR DATA

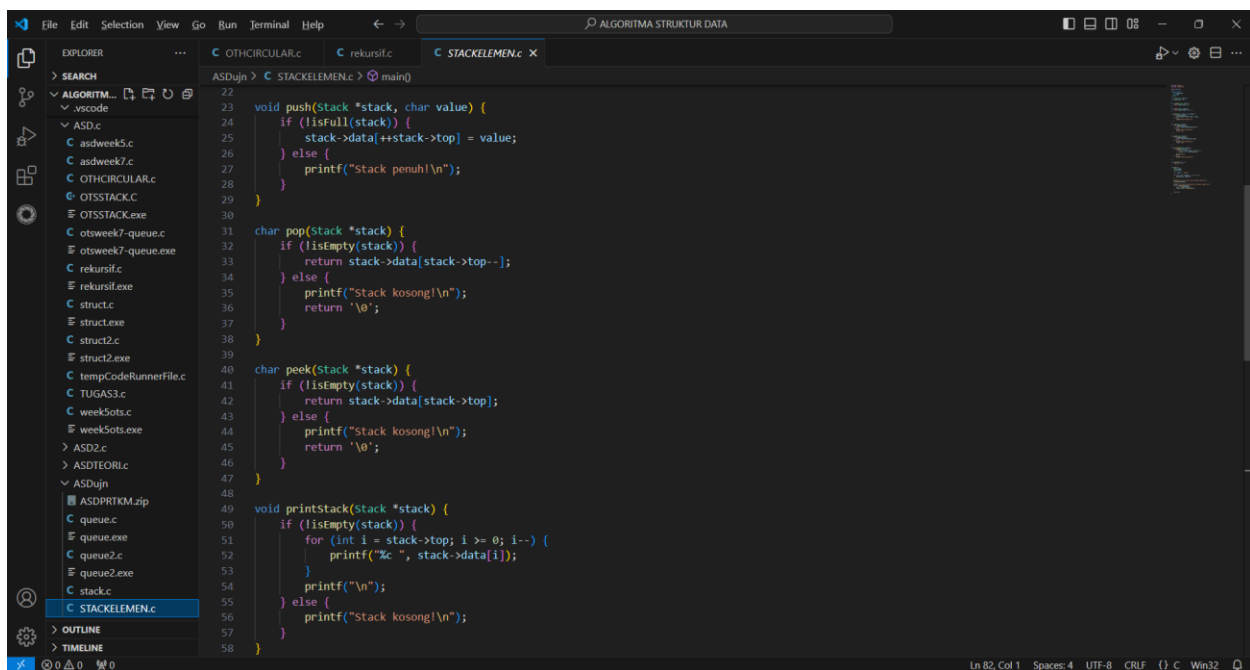
EXPLORER
SEARCH
ALGORITHM...
.vscode
ASD.c
  asdweek5.c
  asdweek7.c
  OTHCIRCULAR.c
  OTSSTACK.c
  OTSSTACK.exe
  otsweek7-queue.c
  otsweek7-queue.exe
  rekursif.c
  rekursif.exe
  struct.c
  struct.exe
  struct2.c
  struct2.exe
  tempCodeRunnerFile.c
  TUGAS3.c
  week5ots.c
  week5ots.exe
  ASD2.c
  ASDTEORI.c
  ASDujn
    ASDPRTKM.zip
    queue.c
    queue.exe
    queue2.c
    queue2.exe
    stack.c
    STACKELEMEN.c
  OUTLINE
  TIMELINE

ASDujn > C stackc > ...
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAX_STACK_SIZE 5
5
6 typedef struct stack {
7     int data[MAX_STACK_SIZE];
8     int top;
9 }stack;
10
11 void printstack(stack*stack){
12     if(isempty(stack)){
13         printf("\nstack kosong" );
14         return;
15     }
16     printf("\nstack:");
17     for (int i= stack ->0; i--){
18         printf("%d", stack ->data[i]);
19     }
20 }
21
```

4.STACK ELEMEN



```
1 #include <stdio.h>
2 #include <stdbool.h>
3
4 #define MAX 10
5
6 typedef struct {
7     char data[MAX];
8     int top;
9 } Stack;
10
11 void init(Stack *stack) {
12     stack->top = -1;
13 }
14
15 bool isEmpty(Stack *stack) {
16     return stack->top == -1;
17 }
18
19 bool isFull(Stack *stack) {
20     return stack->top == MAX - 1;
21 }
22
23 void push(Stack *stack, char value) {
24     if (!isFull(stack)) {
25         stack->data[++stack->top] = value;
26     } else {
27         printf("Stack penuh!\n");
28     }
29 }
30
31 char pop(Stack *stack) {
32     if (!isEmpty(stack)) {
33         return stack->data[stack->top--];
34     } else {
35         printf("Stack kosong!\n");
36         return '\0';
37     }
38 }
39
40 char peek(Stack *stack) {
41     if (!isEmpty(stack)) {
42         return stack->data[stack->top];
43     } else {
44         printf("Stack kosong!\n");
45         return '\0';
46     }
47 }
48
49 void printStack(Stack *stack) {
50     if (!isEmpty(stack)) {
51         for (int i = stack->top; i >= 0; i--) {
52             printf("%c ", stack->data[i]);
53         }
54         printf("\n");
55     } else {
56         printf("Stack kosong!\n");
57     }
58 }
```



```
1 #include <stdio.h>
2 #include <stdbool.h>
3
4 #define MAX 10
5
6 typedef struct {
7     char data[MAX];
8     int top;
9 } Stack;
10
11 void init(Stack *stack) {
12     stack->top = -1;
13 }
14
15 bool isEmpty(Stack *stack) {
16     return stack->top == -1;
17 }
18
19 bool isFull(Stack *stack) {
20     return stack->top == MAX - 1;
21 }
22
23 void push(Stack *stack, char value) {
24     if (!isFull(stack)) {
25         stack->data[++stack->top] = value;
26     } else {
27         printf("Stack penuh!\n");
28     }
29 }
30
31 char pop(Stack *stack) {
32     if (!isEmpty(stack)) {
33         return stack->data[stack->top--];
34     } else {
35         printf("Stack kosong!\n");
36         return '\0';
37     }
38 }
39
40 char peek(Stack *stack) {
41     if (!isEmpty(stack)) {
42         return stack->data[stack->top];
43     } else {
44         printf("Stack kosong!\n");
45         return '\0';
46     }
47 }
48
49 void printStack(Stack *stack) {
50     if (!isEmpty(stack)) {
51         for (int i = stack->top; i >= 0; i--) {
52             printf("%c ", stack->data[i]);
53         }
54         printf("\n");
55     } else {
56         printf("Stack kosong!\n");
57     }
58 }
```

```
49 void printStack(stack *stack) {
50     if (!isEmpty(stack)) {
51         for (int i = stack->top; i >= 0; i--) {
52             printf("%c ", stack->data[i]);
53         }
54         printf("\n");
55     } else {
56         printf("Stack kosong!\n");
57     }
58 }
59
60 int charToInt(char c) {
61     return c - '0';
62 }
63
64 int main() {
65     Stack stack;
66     init(&stack);
67
68     char input[] = "12345";
69
70     for (int i = 0; input[i] != '\0'; i++) {
71         push(&stack, input[i]);
72     }
73
74     printf("Isi stack setelah push karakter angka:\n");
75     printStack(&stack);
76
77     printf("Isi stack setelah konversi menjadi integer:\n");
78     while (!isEmpty(&stack)) {
79         char c = pop(&stack);
80         printf("%d\n", charToInt(c));
81     }
82
83     return 0;
84 }
```