STRUKTUR DATA

QUEUE

Rosa Ariani Sukamto

ROSA ARIANI SUKAMTO

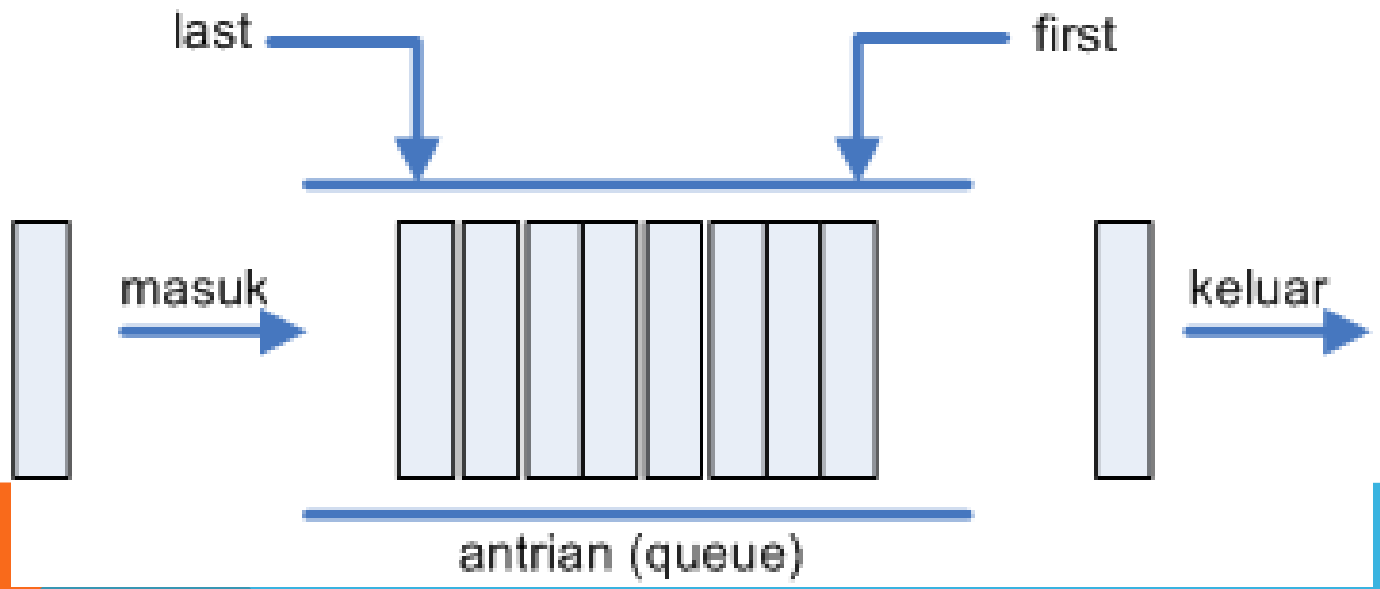
Blog: http://hariiniadalahhadiah.wordpress.com

Facebook: https://www.facebook.com/rosa.ariani.sukamto

Email: rosa_if_itb_01@yahoo.com

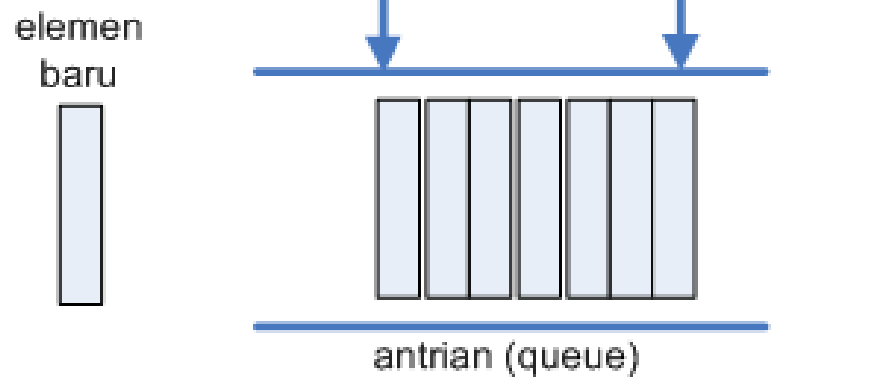
QUEUE

Antrian atau queue (baca : qyu) adalah salah satu konsep struktur data yang memiliki sistem kerja pertama masuk maka akan menjadi yang pertama keluar (FIFO = *First In First Out*) seperti halnya antrian yang ada pada dunia nyata

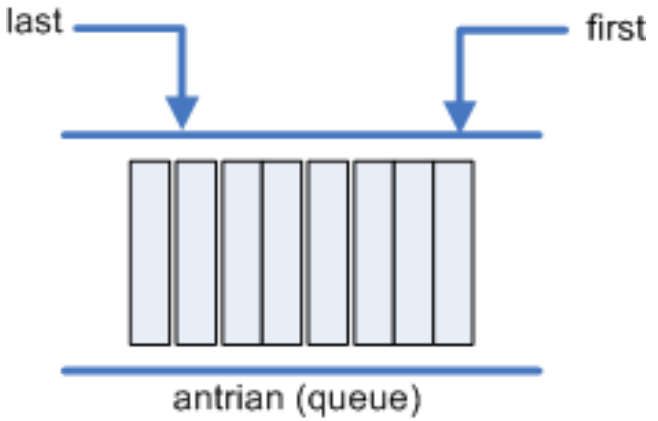


PROSES ADD

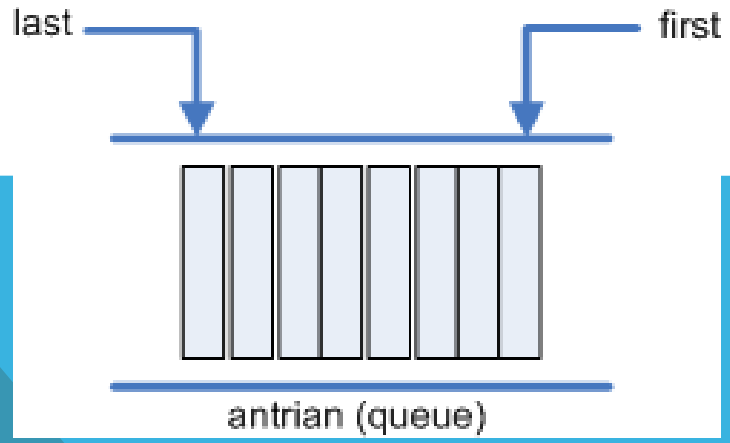
1



2

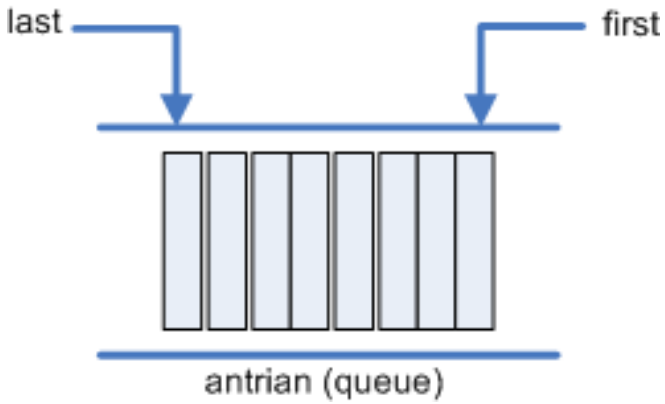


3

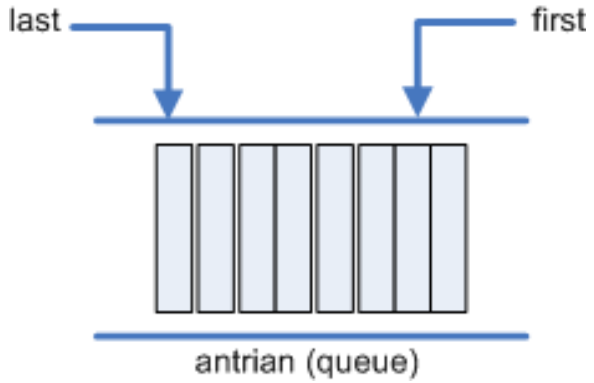


PROSES DEL

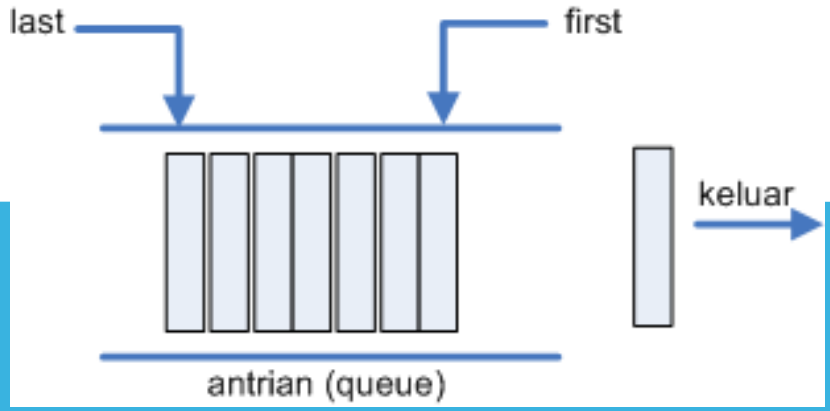
1



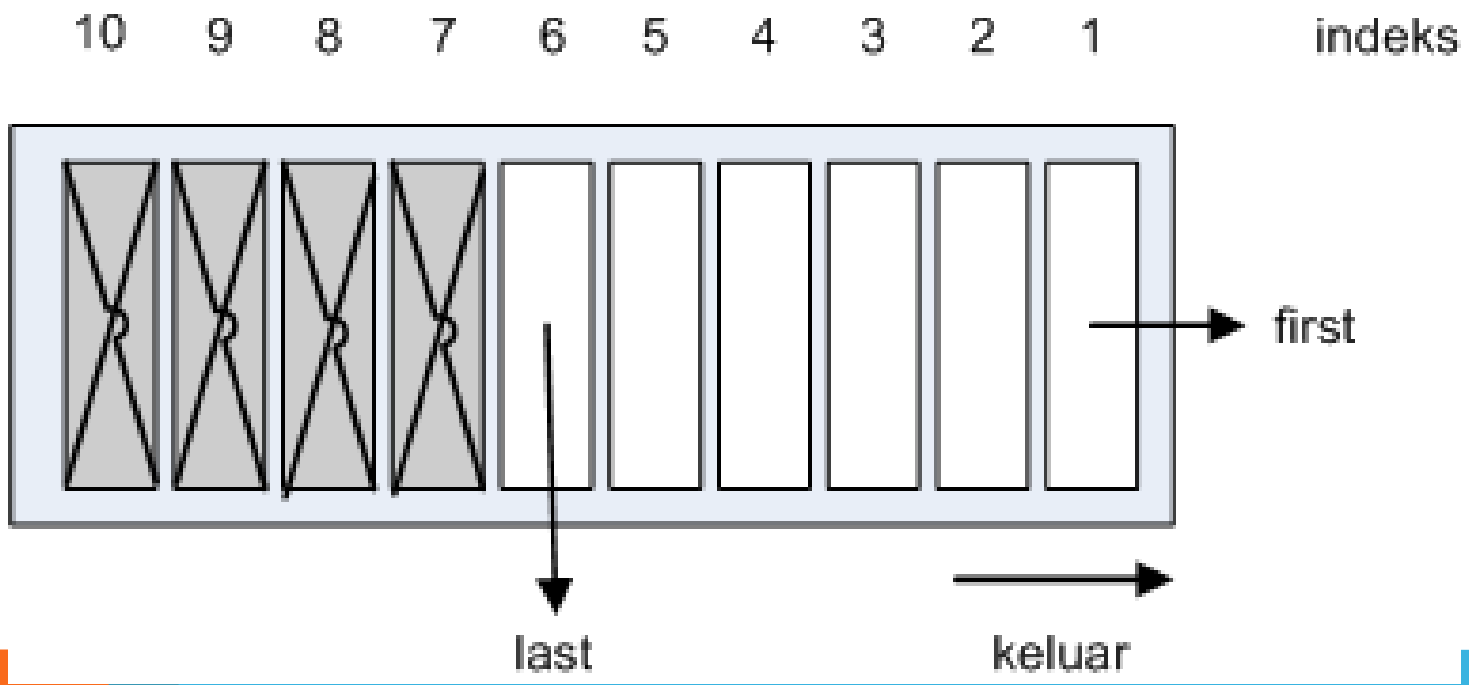
2



3

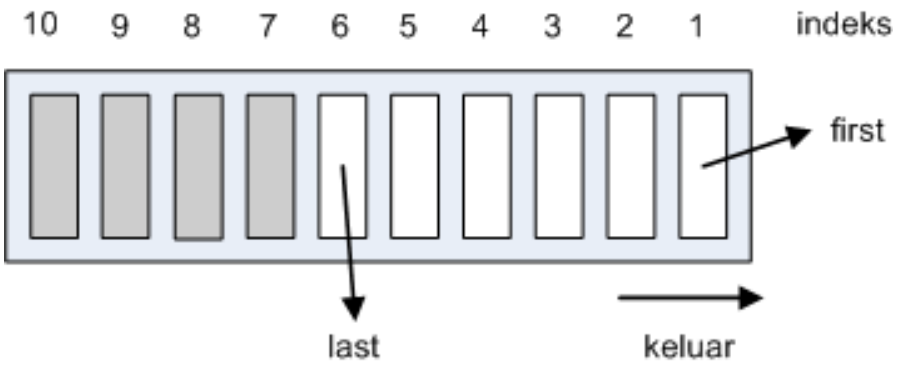


QUEUE REPRESENTASI STATIS

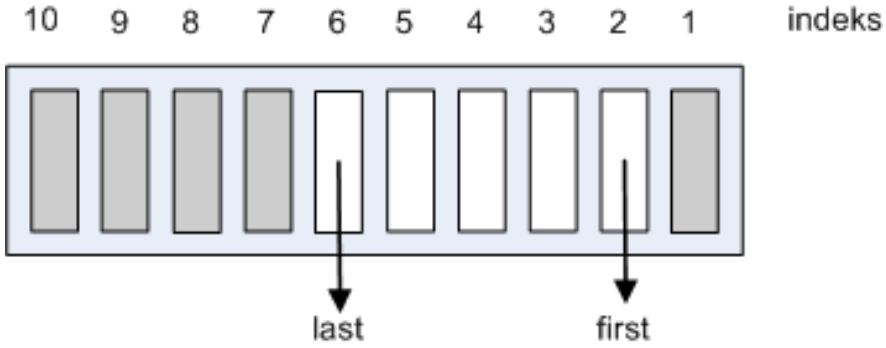


DEL PADA QUEUE STATIS

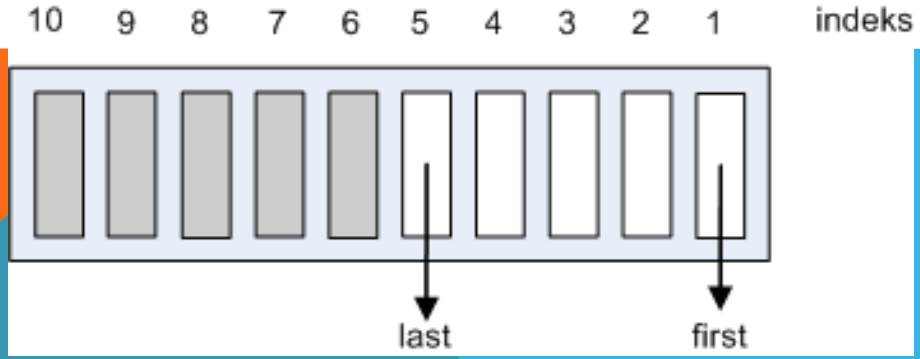
1



2



3



DEKLARASI ELEMEN DAN INISIALISASI

```
#include <stdio.h>
#include <string.h>

typedef struct{
  char nim[10];
  char nama[50];
  float nilai;
}nilaiMatKul;

typedef struct{
  int first;
  int last;
  nilaiMatKul data[10];
}queue;

void createEmpty(queue *Q){
  (*Q).first = -1;
  (*Q).last = -1;
}
```

```
int isEmpty(queue Q){
 int hasil = 0;
 if(Q.first == -1){
   hasil = 1;
 }
 return hasil;
}

int isFull(queue Q){
  int hasil = 0;
  if(Q.last == 9){
    hasil = 1;
  }
  return hasil;
}
```

ADD

```
void add(char nim[], char nama[],
   float nilai, queue *Q ){


  if(isEmpty(*Q) == 1){
    /* jika queue kosong */
    (*Q).first = 0;
    (*Q).last = 0;
    strcpy((*Q).data[0].nim,
   nim);


    strcpy((*Q).data[0].nama,
   nama);


    (*Q).data[0].nilai = nilai;
  }
```

```
  else{
    /* jika queue tidak kosong */
    if(isFull(*Q) != 1){
      (*Q).last = (*Q).last + 1;

   strcpy((*Q).data[(*Q).last].nim,
   nim);


   strcpy((*Q).data[(*Q).last].nama
   , nama);
    (*Q).data[(*Q).last].nilai =
   nilai;
    }
    else{
       printf("queue penuh\n");
    }
  }


}
```

DEL

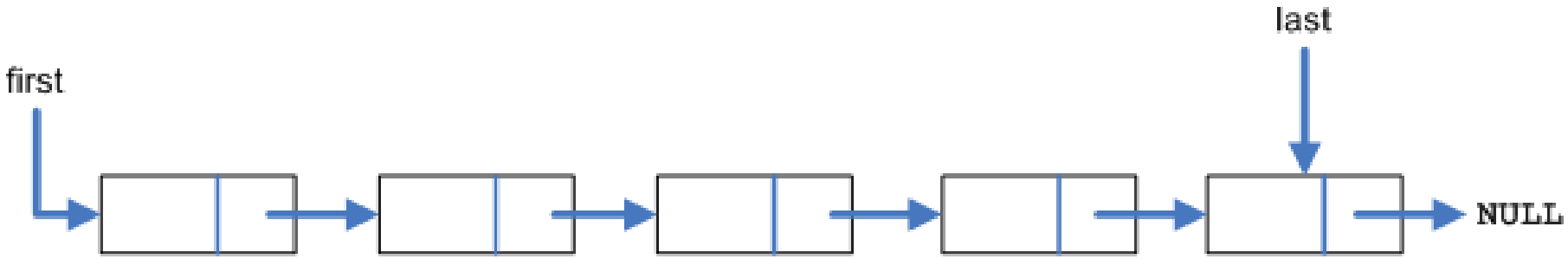
```
void del(queue *Q){


  if((*Q).last == 0){
    (*Q).first = -1;
    (*Q).last = -1;
  }else{
    /*menggeser elemen ke depan*/
    int i;
    for(i=((*Q).first + 1);i<=(*Q).last;i++){
    strcpy((*Q).data[i-1].nim, (*Q).data[i].nim);
    strcpy((*Q).data[i-1].nama,
   (*Q).data[i].nama);
    (*Q).data[i-1].nilai = (*Q).data[i].nilai;
    }
    (*Q).last = (*Q).last - 1;
  }
}
```

PRINT QUEUE DAN MAIN

```
void printQueue(queue Q){
  if(Q.first != -1){
   printf("------isi queue------\n");
   int i;
   for(i=Q.last;i>=Q.first;i--){
    printf("====================\n");
    printf("elemen ke : %d\n", i);
    printf("nim : %s\n",
   Q.data[i].nim);
    printf("nama : %s\n",
   Q.data[i].nama);
    printf("nilai : %f\n",
   Q.data[i].nilai);
   }
   printf("--------------------\n");
  }
  else{
    /* proses jika queue kosong */
    printf("queue kosong\n");
  }
}
```

```
int main(){
  queue Q;
  createEmpty(&Q);
  printQueue(Q);

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

  add("13507701", "Nana", 64.75, &Q);
  add("13507702", "Rudi", 75.11, &Q);
  add("13507703", "Dea", 84.63, &Q);
  printQueue(Q);

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

  del(&Q);
  del(&Q);
  printQueue(Q);

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

  return 0;
}
```

QUEUE REPRESENTASI DINAMIS



DEKLARASI ELEMEN DAN INISIALISASI

```
#include <stdio.h>
#include <malloc.h>
#include <string.h>

typedef struct{
  char nim[10];
  char nama[50];
  float nilai;
}nilaiMatKul;

typedef struct elm *alamatelmt;
typedef struct elm{
  nilaiMatKul elmt;
  alamatelmt next;
}elemen;

typedef struct{
  elemen *first;
  elemen *last;
}queue;
```

```
void createEmpty(queue *Q){
   (*Q).first = NULL;
   (*Q).last = NULL;
}


int isEmpty(queue Q){
 int hasil = 0;
 if(Q.first == NULL){
    hasil = 1;
 }
 return hasil;
}
```

COUNTELEMENT

```
int countElement(queue Q){

  int hasil = 0;

  if(Q.first != NULL){
    /* queue tidak kosong */

    elemen *elmt;

    /* inisialisasi */
    elmt = Q.first;

    while(elmt != NULL){
      /* proses */
      hasil= hasil + 1;

      /* iterasi */
      elmt = elmt->next;
    }

  }

  return hasil;

}
```

ADD

```
void add(char nim[], char nama[], float nilai, queue *Q ){

  elemen *elmt;
  elmt = (elemen *) malloc (sizeof (elemen));
  strcpy(elmt->elmt.nim, nim);
  strcpy(elmt->elmt.nama, nama);
  elmt->elmt.nilai = nilai;
  elmt->next = NULL;
  if((*Q).first == NULL){
      (*Q).first = elmt;
  }
  else{
      (*Q).last->next = elmt;
  }
  (*Q).last = elmt;
  elmt = NULL;

}
```

DEL

```
void del(queue *Q){


  if((*Q).first != NULL){
  /* jika queue bukan queue kosong */


    elemen *elmt = (*Q).first;
    (*Q).first = (*Q).first->next;
    elmt->next = NULL;
    free(elmt);
  }


}
```

PRINTQUEUE

```
void printQueue(queue Q){

  if(Q.first != NULL){
   printf("------isi queue------
   \n");

   elemen *elmt = Q.first;

   int i = 1;

   while(elmt != NULL){

   printf("====================\n")
   ;
    printf("elemen ke : %d\n", i);
    printf("nim : %s\n",
      elmt->elmt.nim);
    printf("nama : %s\n",
      elmt->elmt.nama);
    printf("nilai : %f\n",
      elmt->elmt.nilai);

      /* iterasi */
      elmt = elmt->next;
      i = i + 1;

   }
   printf("--------------------
   \n");
  }
  else{
    /* proses jika queue kosong
   */
    printf("queue kosong\n");
  }

}
```

MAIN

```
int main(){
  queue Q;
  createEmpty(&Q);
  printQueue(Q);
  printf("================\n");

  add("13507701", "Nana", 64.75, &Q);
  add("13507702", "Rudi", 75.11, &Q);
  add("13507703", "Dea", 84.63, &Q);
  printQueue(Q);

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

  del(&Q);
  del(&Q);
  printQueue(Q);

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

  return 0;
}
```

QUEUE BERPRIORITAS

- **Masukan ditambah dengan prioritas**
- **Kondisi kasus**
- Di depan
  - Geser semua ke belakang agar tempat pertama dapat ditempati elemen baru
- Di tengah
  - Geser dari posisi yang diinginkan sampai ke belakang agar tempat posisi yang diinginkan dapat diisi oleh elemen baru
- Di belakang
  - Masukkan elemen baru di belakang

DAFTAR PUSTAKA

S, Rosa A. dan M. Shalahuddin. 2010. Modul Pembelajaran: Struktur Data. Modula: Bandung.