**Queue**

Sebuah mesin di pabrik menjalakan tugas secara urut dan terus menerus. Masing-masing tugas memiliki waktu penyelesaian yang berbeda-beda. Ketika tiap tugas selesai dikerjakan maka tugas tersebut akan masuk kembali ke dalam antrian tugas paling belakang dan statusnya berubah menjadi “finished”. Urutan tugas mesin direpresentasikan dengan ***circular*** *single-linked-list*seperti berikut:



Status: unfinished

Status: unfinished

Status: unfinished

Status: unfinished

ADT:

**type** infotype : <

    Pekerjaan      : **char**  , *{ID pekerjaan}*

    waktu : **integer**   *{waktu yang dibutuhkan untuk menyelesaikan pekerjaan}*

Dikerjakan*:***integer***{waktu yang telah digunakan untuk mengerjakan pekerjaan tersebut}*

>

**type** address : **pointer** **to** ElmTugas

**type** ElmTugas : < info : infotype , next : address>

**type** AntrianTugas : < Head, Tail : address >

1. Buatlah algoritma untuk menambahkan data tugas ke dalam antrian

|  |
| --- |
| **procedure** TambahTugas (**Input/Output** Q: ………………., **Input** P : ……………..)  *{I.S. Terdefinisi antrian Q mungkin kosong dan alamat P yang menunjuk sebuah pekerjaan*  *F.S. P berada di Tail}* |
| **Kamus**  **X = new AntrianTugas Q = queue AntrianTugas P : pointer to addres** |
| **Algoritma**  **if (tail==head) & (Q==null) {  head 🡸 X  tail 🡸 X } Else {  Next(p) 🡸 X Tail 🡸 X }** |
|  |

1. Dengan menggunakan procedure/function yang sudah tersedia di atas, buatlah procedure yang menjalankan semua tugas dari paling depan sampai seluruh tugas berubah statusnya menjadi “finished”, kemudian total waktu pengerjaan tugas ditampilkan di layar. Ilustrasinya akan menjadi seperti berikut:



Status: finished

Status: finished

Status: finished

Status: finished

**Output:**

**Total = 22 jam**

|  |
| --- |
| **Procedure** DailyTask **(Input/Output**Q: ……………………….**)** *{I.S. Terdefinisi queue Q****tidak kosong***  *{F.S. Ada elemen-elemen yang telah dihapus yg merepresentasikan tugas yang telah diproses dan diantrikan kembali dari belakang}* |
| **Kamus**  **procedure** HapusTugas (**Input/Output** Q: ……………………….., **Output** P : ………………….)  **procedure** TambahTugas (**Input/Output** Q: ……………….., **Input** P : …………………….)  **X = new AntrianTugas Q = queue AntrianTugas P : pointer to addres** |
| **Algoritma  int jumlah  while (P!=null){ if (P==Head){ X 🡸 Head Head 🡸 HapusTugas(Head) P🡸 Head }else if(P==Tail){  TambahTugas(X) } else {  jumlah += P.info.Dikerjakan  p 🡸 Next(P) }}** |
|  |