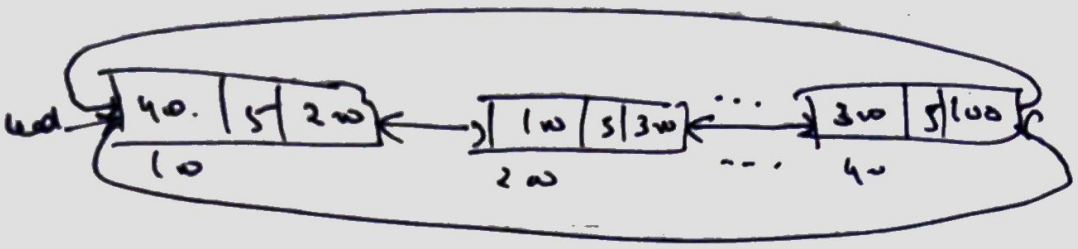
Double Circular Linked List




```
Struct node
{
    int data;
    s n * next;
    s n* prv;
};
main()
{
    sn *head = NULL;
    create(&head);
    disp(head);
}
```

```
void create( int n, sn **h)
{
    int i;
    sn *cur, *ptr;
    for(i=0; i<n; i++)
    {
        cur = (sn*) malloc(_____);
        cur->data = random();
        cur->next = cur->prev = NULL;

        if(*h == NULL)
        {
            *h = cur;
            ptr = cur;
            ptr->next = *h;
            ptr->prev = *h;
        }
        else
        {
            ptr->next = cur;
            cur->next = *h;
            (*h)->prev = cur;
            ptr = ptr->next;
        }
    }
}
```

cur->prev = ptr;
cur->next = ptr->next;
ptr->next = cur;
(*h)->prev = ptr->next;

```
void insert( sn **ah, int v, int p)
{
    sn *cur, *ptr;
    cur = (sn *)malloc( ... );
    cur->data = v;
    cur->next = cur->prev = NULL;
    if (*ah == NULL)
    {
        *ah = cur;
        cur->next = cur->prev = cur;
    }
    else if ( p == 0)
    {
        cur->next = *ah;
        cur->prev = (*ah)->prev;
        (*ah)->prev->next = cur;
        (*ah)->prev = cur;
        *ah = cur;
    }
```

```
else
{
    ptr = 4th;
    int i = 1;
    while( i<((p-1) % length()+1)
    {
        ptr = ptr->next;
        i++;
    }
    cur->prev = ptr;
    cur->next = ptr->next
    ptr->next->prev = cur
    ptr->next = cur;
}
}
```

```
void del(sn **h, int p)
{
    sn *ptr;
    if (*h == NULL)
        printf("Empty");
    else
    {
        ptr = *h;
        i = 1;
        while(i<p && ptr->next != *h)
        {
            ptr = ptr->next;
            i++;
        }
        if(ptr->next == *h && i<p)
            printf("Not found");
        else if(ptr == *h && ptr->next == ptr)
        {
            *h = NULL;
            free(ptr);
        }
    }
```

```
else if (ptr == ah)
{
    ptr -> prev -> next = ptr -> next;
    ptr -> next -> prev = ptr -> prev;
    ah = (ah) -> next;
    free (ptr);
}
else
{
    ptr -> prev -> next = ptr -> next;
    ptr -> next -> prev = ptr -> prev;
    free (ptr);
}
}
```