

# 202251142\_semaphore

## Code

---

```
// Kunj Thakkar
// 202251142

#include<stdio.h>
#include<stdlib.h>

struct node {
    int pid;
    struct node* next;
};

struct queue{
    struct node* front;
    struct node* rear;
};

struct queue* createQueue(){
    struct queue* q = (struct queue*)malloc(sizeof(struct queue));
    q->front = q ->rear = NULL;
    return q;
}

void enqueue(struct queue* q, int pid){
    struct node* n = (struct node*)malloc(sizeof(struct node));
    n -> pid = pid;
    n -> next = NULL;
    if(q -> rear == NULL){
        q -> front = q -> rear = n;
    }
    else{
        q -> rear -> next = n;
        q -> rear = n;
    }
}
```

```

    return ;
}

q -> rear -> next = n;

return;
}

int dequeue(struct queue* q){

if(q -> front == NULL){

return -1;

}

struct node* n = q -> front;

int id = n -> pid;

q -> front = q -> front -> next;





if(q -> front == NULL){

q -> rear = NULL;

}

free(n);

return id;

}

void wait (int* s, struct queue* q, int id){

(*s)--;

if(*s < 0){

enqueue(q,id);

printf("process %d has entered queue\n", id);

}

else{

printf("process %d continues to execute\n", id);

}

}

```

```
void signal(int *s, struct queue* q){  
    (*s)++;  
  
    if(*s <= 0){  
  
        int id = dequeue(q);  
  
        if(id!= -1) printf("process %d is ready for execution\n", id);  
  
    }  
  
    else printf("no process is to execute \n");  
  
}  
  
int main(){  
  
    int s = 1;  
  
    struct queue* q = createQueue();  
  
    int p1 = 1;  
  
    int p2 = 2;  
  
    int p3 = 3;  
  
  
  
    wait(&s, q, p1); // s = 0  
  
    wait(&s, q, p3); // s = -1  
  
    wait(&s, q, p2); // s = -2  
  
    signal(&s,q); // s = -1  
  
    signal(&s,q); // s = 0  
  
    wait(&s, q, p1); // s = -1  
  
    wait(&s, q, p2); // s = -2  
  
    signal(&s,q); // s = -1  
  
  
  
    return 0;  
}
```

## output

---

```
process 1 continues to execute
process 3 has entered queue
process 2 has entered queue
process 3 is ready for execution
process 2 is ready for execution
process 1 has entered queue
process 2 has entered queue
process 1 is ready for execution
```