202135574 Jeon Sihyeon (전시현)

PHW1-1

텍스트이(가) 표시된 사진

자동 생성된 설명

#include <stdio.h>

int bufferfull();

int empty();

int insert(int num);

int delete();

int ring\_buffer[5];

int rear = 0;

int front = 0;

int main(){

    int del\_res;

    int insert\_res;

    //1

    printf("Initially, Rear = %d, Front = %d\n", rear, front);

    int num;

    //2

    num = 10;

    insert\_res = insert(num);

    if (insert\_res == 0){

        printf("failed to insert %d\n", num);

    }

    else{

        printf("Insert %d, Rear = %d, Front = %d\n", num, rear, front);

    }

    //3

    num = 50;

    insert\_res = insert(num);

    if (insert\_res == 0){

        printf("failed to insert %d\n", num);

    }

    else{

        printf("Insert %d, Rear = %d, Front = %d\n", num, rear, front);

    }

    //4

    num = 20;

    insert\_res = insert(num);

    if (insert\_res == 0){

        printf("failed to insert %d\n", num);

    }

    else{

        printf("Insert %d, Rear = %d, Front = %d\n", num, rear, front);

    }

    //5

    num = 70;

    insert\_res = insert(num);

    if (insert\_res == 0){

        printf("failed to insert %d\n", num);

    }

    else{

        printf("Insert %d, Rear = %d, Front = %d\n", num, rear, front);

    }

    //6

    del\_res = delete();

    if (del\_res == 0){

        printf("failed to delete\n");

    }

    else{

        printf("Delete front, Rear = %d, Front = %d\n", rear, front);

    }

    //7

    num = 100;

    insert\_res = insert(num);

    if (insert\_res == 0){

        printf("failed to insert %d\n", num);

    }

    else{

        printf("Insert %d, Rear = %d, Front = %d\n", num, rear, front);

    }

    //8

    num = 40;

    insert\_res = insert(num);

    if (insert\_res == 0){

        printf("failed to insert %d\n", num);

    }

    else{

        printf("Insert %d, Rear = %d, Front = %d\n", num, rear, front);

    }

    //9

    num = 140;

    insert\_res = insert(num);

    if (insert\_res == 0){

        printf("failed to insert %d, Rear = %d, front = %d.\nAs Front = Rear + 1, so Queue overflow\n", num, rear, front);

    }

    else{

        printf("Insert %d, Rear = %d, Front = %d\n", num, rear, front);

    }

    //10

    del\_res = delete();

    if (del\_res == 0){

        printf("failed to delete\n");

    }

    else{

        printf("Delete front, Rear = %d, Front = %d\n", rear, front);

    }

    //11

    del\_res = delete();

    if (del\_res == 0){

        printf("failed to delete\n");

    }

    else{

        printf("Delete front, Rear = %d, Front = %d\n", rear, front);

    }

    //12

    del\_res = delete();

    if (del\_res == 0){

        printf("failed to delete\n");

    }

    else{

        printf("Delete front, Rear = %d, Front = %d\n", rear, front);

    }

    return 0;

}

int bufferfull(){

    if ((front-rear-1)%5==0){

        return 1;   //full

    }

    return 0;   // not full

}

int empty(){

    if (front==rear&&ring\_buffer[rear]==NULL){

        return 1;       //empty

    }

    return 0;       //not empty

}

int insert(int num){

    if (bufferfull()==1){

        return 0;   //failed to insert num

    }

    if (rear/5!=0){

        rear%=5;

    }

    if (front/5!=0){

        front%=5;

    }

    ring\_buffer[(rear+1)%5] = num;

    if(empty()==1){

        rear++;

        front++;

    }

    else

        rear++;

    return 1;       //success to insert num

}

int delete(){

    if (empty()==1){

        return 0;   //failed to delete num

    }

    if (rear/5!=0){

        rear%=5;

    }

    if (front/5!=0){

        front%=5;

    }

    ring\_buffer[(front+1)%5] = NULL;

    front++;

    return 1;       //success to delete num

}

PHW 1-4

텍스트이(가) 표시된 사진

자동 생성된 설명

#include <stdio.h>

void print\_queue();

int queue\_full();

int queue\_empty();

int dequeue1Nenqueue2();

int dequeue1();

void enqueue2(int num);

int dequeue2Nenqueue1();

int dequeue2();

void enqueue1(int num);

void pop\_rear();

int pop();

int push(int num);

#define MAX 10

int queue1[10], queue2[10];

int rear1 = -1, rear2 = -1;

int front1 = -1, front2 = -1;

int main(){

    print\_queue();

    int push\_res;

    for (int i = 1; i<6; i++){

        push\_res = push(i);

        if (push\_res==0){

            printf("failed to push %d -- (The queue is full)\n", i);

        }

        print\_queue();

        printf("       -- successfully pushed %d\n", i);

    }

    int pop\_res;

    pop\_res = pop();

    print\_queue();

    if (pop\_res == 0){

        printf("failed to pop %d (rear)\n", queue1[rear1]);

    }

    else{

        printf("       -- successfully popped the top\n");

    }

    for (int i = 1; i<10; i++){

        push\_res = push(i);

        if (push\_res==0){

            printf("failed to push %d -- (The queue is full)\n", i);

        }

        print\_queue();

        printf("       -- successfully pushed %d\n", i);

    }

    for (int i = 0; i<12; i++){

        pop\_res = pop();

        if (pop\_res == 0){

            printf("failed to pop -- (The queue is empty)\n");

        }

        print\_queue();

        printf("       -- successfully popped the top\n");

    }

    return 0;

}

void print\_queue(){

    //printf("first : ");

    for (int i = 0; i<MAX; i++){

        if(queue1[i]==NULL){

            break;

        }

        printf("%d ", queue1[i]);

    }

    //printf("\n");

    //printf("second : ");

    for (int i = 0; i<MAX; i++){

        if(queue2[i]==NULL){

            break;

        }

        //printf("%d ", queue2[i]);

    }

    //printf("\n");

    //printf("rear1 : %d, front1 : %d, rear2 : %d, front2 : %d\n", rear1, front1, rear2, front2);

}

int queue\_full(){

    if (rear1 == 9){

        return 1;   //  queue1 is full

    }

    if (rear2 == 9){

        return 2;   //  queue2 is full

    }

    return 0;       //  none of them is full

}

int queue\_empty(){

    if (rear1 == -1){

        return 1;   // queue1 is empty

    }

    return 0;       //none of them is empty

}

int dequeue1Nenqueue2(){

    if (queue\_empty()==1){

        return 0;   //failed

    }

    int num = dequeue1();

    enqueue2(num);

    return 1;       //success

}

int dequeue1(){

    int num;

    num = queue1[front1];

    queue1[front1] = NULL;

    if (rear1 >= front1+1){

        front1++;

    }

    return num;

}

void enqueue2(int num){

    if (front2 == -1){

        front2++;

    }

    rear2++;

    queue2[rear2] = num;

}

int dequeue2Nenqueue1(){

    int num = dequeue2();

    enqueue1(num);

    return 1;       //success

}

int dequeue2(){

    int num;

    num = queue2[front2];

    queue2[front2] = NULL;

    if (rear2>=front2+1){

        front2++;

    }

    //printf("\nnum : %d front1 : %d front2 : %d\n", num, front2);

    return num;

}

void enqueue1(int num){

    if (front1==-1){

        front1++;

    }

    rear1++;

    queue1[rear1] = num;

    //printf(" queue 1[rear1] : %d num : %d\n", queue1[rear1], num);

}

void pop\_rear(){

    queue1[front1] = NULL;

}

int pop(){

    int res1 = dequeue1Nenqueue2();

    if (res1==0){

        return 0;   //failed

    }

    if (front1==0){

        queue1[front1] = NULL;

        front1--;

        rear1--;

        return 1;

    }

    int temp;

    //printf("res1 : %d rear1 : %d\n-----\n", res1, rear1);

    for (int i = 0; i<rear1-1; i++){

        temp = dequeue1Nenqueue2();

    }

    pop\_rear();

    rear1 = -1;

    front1 = -1;

    for (int i = 0; i<rear2+1; i++){

        temp = dequeue2Nenqueue1();

        //print\_queue();

    }

    initialize\_queue2();

    return 1;

}

int push(int num){

    if (queue\_full()!=1){

        if (rear1 == -1){

            front1++;

        }

        queue1[++rear1] = num;

        return 1;       //success

    }

    else

        return 0;       //failed

}

void initialize\_queue2(){

    for (int i = 0; i<MAX; i++){

        queue2[i] = NULL;

    }

    rear2 = -1;

    front2 = -1;

}