ルビス　ルスファン　アンシャー

15\_15953

課題（Queueの実装）

プログラム名：queue.c

プログラムの中、nodeという構造体が宣言された。

struct node{

int val;

struct node \*next; //連結リストになる

};

プログラムは5つの関数から成り立つ。

put関数は先頭にデータを追加する。

int put(struct node \*head, int val){

//head->nextに代入するための新しいアドレスを作成

struct node \*new;

new = (struct node \*)malloc(sizeof(struct node));

if (new == NULL) return -1;

if (head -> val == -1){

//headに{-1}のある場合、-1を追加したいデータと交換

head -> val = val;

head -> next = NULL;

} else {

//そうではない場合、新しいアドレスにheadのvalとnextを代入し、headのアドレスに追加したいデータのvalとnextを入れる。そうすると、headのアドレスは不変。

new -> val = head -> val;

new -> next = head -> next;

head -> val = val;

head -> next = new;

}

return 0;

}

get関数は最初にqueueに入ったval、つまり末尾にあるものを取り出して、そのアドレスをqueueから削除する。

int get(struct node \*head){

struct node \*t = head;

if (t == NULL) return -1;

//(t -> next) -> nextはNULLになるまで、ループをする

while((t -> next) -> next != NULL){

t = t -> next;

}

int p = (t -> next) -> val; // (t -> next) -> valが求めた値

t -> next = NULL;

free(t -> next);

return p;

}

delete関数はある整数をqueueの中に探し、見つければ出力し、そのアドレスをqueueから削除する。

int delete(struct node \*head, int val){

struct node \*s;

struct node \*t = head;

if (t == NULL) return -1;

//val is in head

//求めた値はheadにある場合、head -> nextをheadにする。headをfreeし、head -> valを出力する.

if (t -> val == val){

head = t -> next;

free(t);

return val;

}

while(t -> next != NULL && t -> val != val){

//sはtの１個まえのアドレスを指す

s = t;

t = t -> next;

}

if (t -> val == val){

//求めた値が見つければ、s -> nextにt -> nextを入れて、tをfree、求めた値を出力として出す

s -> next = t -> next;

free(t);

return val;

} else return -1; //見つからなければ、-1をreturnする

}

display関数はqueueにある要素を末尾から先頭までの順場に表示する。

void display(struct node \*head){

struct node \*t = head;

if(head -> next == NULL){

printf("%d ", head -> val);

} else {

//再起的に表示する

display(head -> next);

printf("%d ", head -> val); //先頭を後ろに表示する

}

}

int main(){

struct node head = {-1, NULL};

int nums[] = {0, 1, 2, 3, 4, 5, 43, 7, 8, 9};

int i, res;

for(i = 0; i < 10; i++){

printf("put %d\n", nums[i]);

res = put(&head, nums[i]);

if(res != 0) return 1;

}

printf("queue: ");

display(&head);

printf("\n");

for (i = 0; i < 3; i++){

res = get(&head);

printf("get %d\n", res);

}

printf("queue: ");

display(&head);

printf("\n");

res = delete(&head, 6);

printf("delete %d\n", res);

printf("queue: ");

display(&head);

printf("\n");

}

main関数では、以下のような例を実行した：

|  |  |
| --- | --- |
| ソースコード | 実行結果 |
| struct node head = {-1, NULL};  int nums[] = {0, 1, 2, 3, 4, 5, 43, 7, 8, 9};  int i, res;  for(i = 0; i < 10; i++){  printf("put %d\n", nums[i]);  res = put(&head, nums[i]);  if(res != 0) return 1;  } | put 0  put 1  put 2  put 3  put 4  put 5  put 43  put 7  put 8  put 9 |
| printf("queue: ");  display(&head);  printf("\n"); | queue: 0 1 2 3 4 5 43 7 8 9 |
| for (i = 0; i < 3; i++){  res = get(&head);  printf("get %d\n", res);  } | get 0  get 1  get 2 |
| printf("queue: ");  display(&head);  printf("\n"); | queue: 3 4 5 43 7 8 9 |
| res = delete(&head, 6);  printf("delete %d\n", res); | delete -1 |
| printf("queue: ");  display(&head);  printf("\n"); | queue: 3 4 5 43 7 8 9 |

|  |  |
| --- | --- |
| ソースコード | 実行結果 |
| struct node head = {-1, NULL};  int nums[] = {0, 1, 2, 3, 4, 5, 43, 7, 8, 9};  int i, res;  for(i = 0; i < 10; i++){  printf("put %d\n", nums[i]);  res = put(&head, nums[i]);  if(res != 0) return 1;  } | put 0  put 1  put 2  put 3  put 4  put 5  put 43  put 7  put 8  put 9 |
| printf("queue: ");  display(&head);  printf("\n"); | queue: 0 1 2 3 4 5 43 7 8 9 |
| for (i = 0; i < 3; i++){  res = get(&head);  printf("get %d\n", res);  } | get 0  get 1  get 2 |
| printf("queue: ");  display(&head);  printf("\n"); | queue: 3 4 5 43 7 8 9 |
| res = delete(&head, 7);  printf("delete %d\n", res); | delete 7 |
| printf("queue: ");  display(&head);  printf("\n"); | queue: 3 4 5 43 8 9 |