**SPRAWOZDANIE**

|  |  |  |  |
| --- | --- | --- | --- |
| **Przedmiot** | Wprowadzenie do Informatyki | **Zadanie** | 5.1 |
| **Autor** | Andrii Godlevskyi | **Grupa** | WCY23KY1S1 |
| **Temat** | Struktury dynamiczne​ | | |

1.Treść zadania

Należy napisać program realizujący operacje

Kolejki (FIFO) zaimplementowanej wskaźnikowo:

Front()– odczytuje element z początku kolejki

Enqueue/Push(el)– wstawia element na koniec kolejki

Dequeue/Pop()– usuwa element z początku kolejki

IsEmpty()

Clear()

Listowanie elementów kolejki

1. Metoda realizacji

Tworzymy kolejkę i funkcje operujące na nią i w zależności od wyboru użytkownika wywołujemy potrzebną funkcje i wyprowadzamy wynik na ekran.

1. Założenia / ograniczenia dotyczące danych:

1. Dane wejściowe

Dane obiektów – wczytywane z klawiatury

1. Dane wyjściowe

Dane zapisanych obiektów - wyświetlone na ekranie *(z opisami).*

1. Realizacja
2. Algorytm

Изображение выглядит как текст, рукописный текст, бумага, Бумажное изделие

Автоматически созданное описание

1. Kod zródłowy

#include <stdio.h>

#include <stdlib.h>

//deklaracja struktury

struct Node {

int data;

struct Node\* next;

};

//deklaracja kolejki

struct Queue {

struct Node \*front, \*back;

};

//funkcja inicjalizowania kolejki

void initQueue(struct Queue\* q) {

q->front = q->back = NULL;

}

//funkcja dodania elementu

void Pushel(struct Queue\* q, int value) {

struct Node\* temp = (struct Node\*)malloc(sizeof(struct Node));

temp->data = value;

temp->next = NULL;

if (q->back == NULL) {

q->front = q->back = temp;

return;

}

q->back->next = temp;

q->back = temp;

}

//funkcja znisczenia ostatniego elementu

void Pop(struct Queue\* q) {

if (q->front == NULL) {

printf("Queue is empty\n");

return;

}

struct Node\* temp = q->front;

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

free(temp);

if (q->front == NULL) {

q->back = NULL;

}

}

//funkcja sprawdzenia czy kolejka jest pusta

int IsEmpty(struct Queue\* q) {

return q->front == NULL;

}

//funkcja wyczysczenia kolejki

void Clear(struct Queue\* q) {

while (!IsEmpty(q)) {

Pop(q);

}

}

//funkcja wypisania wszystkich elementow

void ListElements(struct Queue\* q) {

struct Node\* temp = q->front;

while (temp != NULL) {

printf("%d ", temp->data);

temp = temp->next;

}

printf("\n");

}

int main() {

//deklaracja zmiennych

struct Queue q;

initQueue(&q);

//menu wyboru

int wybor, value;

do {

printf("\n1. Dodaj\n2. Pop\n3. Czy pusta?\n4. Oczysc\n5. Lista elementow\n0. Wyjscie\n");

scanf("%d", &wybor);

switch (wybor) {

case 1:

printf("Wpisz wartosc do dodania: ");

scanf("%d", &value);

Pushel(&q, value);

break;

case 2:

Pop(&q);

break;

case 3:

if (IsEmpty(&q)) {

printf("Kolejka jest pusta\n");

} else {

printf("Kolejka nie jest pusta\n");

}

break;

case 4:

Clear(&q);

printf("Kolejka oczyszczona\n");

break;

case 5:

ListElements(&q);

break;

case 0:

printf("Wyjscie z programu\n");

break;

default:

printf("Zly wybor,sprobuj ponownie\n");

}

} while (wybor != 0);

return 0;

}

3. Dane wejściowe /wyjściowe

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, меню

Автоматически созданное описание

1. Złożoność obliczeniowa algorytmu

O(n)