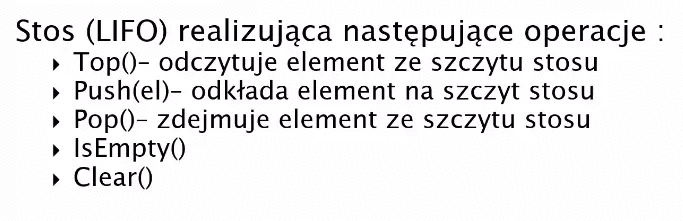
**SPRAWOZDANIE**

|  |  |  |  |
| --- | --- | --- | --- |
| **Przedmiot** | Wprowadzenie do Informatyki | **Zadanie** | 2 |
| **Autor** | Mateusz Jasiński | **Grupa** | WCY20IY2S1 |
| **Temat** | Struktury dynamiczne – kolejki i stosy | | |

1. Treść
   1. Metoda realizacji



* 1. Założenia / ograniczenia dotyczące danych:

W zależności od wybranego polecenia:

-Odczytać element, na szczycie stosu,  
-Dodać element na szczyt stosu,  
-Usunąć sztyt stosu,  
-Sprawdzić czy stos jest pusty,  
-Usunąć wszystkie elementy stosu.

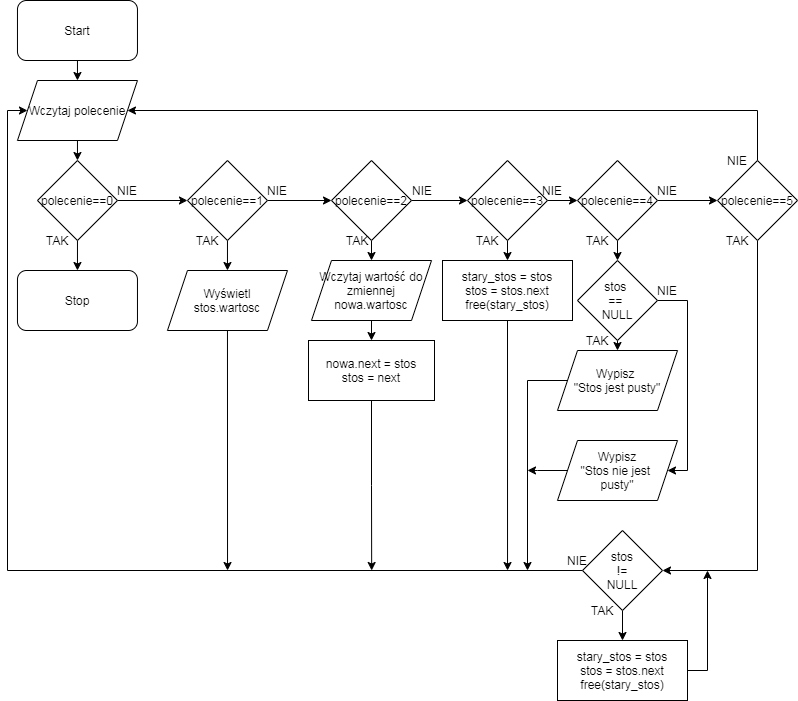
* + 1. Dane wejściowe

Dane obiektów – wprowadzone z klawiatury

* + 1. Dane wyjściowe

Dane zapisanych obiektów – wyprowadzone na ekran

1. Realizacja
   1. Algorytm



* 1. Kod źródłowy

#include<stdio.h>

#include<stdlib.h>

typedef struct lifo {

int wartosc;

struct lifo \*nastepny;

} lifo;

int gui() {

system("cls");

printf("Kolejka LIFO\n"

"----------------------------------------------\n"

"[0] Wyjscie z programu\n"

"[1] Top()\n"

"[2] Push(el)\n"

"[3] Pop()\n"

"[4] IsEmpty()\n"

"[5] Clear()\n"

"----------------------------------------------\n\n"

"Wybrana komenda: ");

}

int IsEmpty(lifo \*kolejka) {

if(kolejka==NULL)

return 1;

else

return 0;

}

Top(lifo \*kolejka) {

if(IsEmpty(kolejka))

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

else

printf("%d\n",kolejka->wartosc);

printf("\nKliknij aby kontynuowac.");

getch();

}

Push(lifo \*\*kolejka) {

lifo \*nowy;

nowy=(lifo\*)malloc(sizeof(lifo));

nowy->nastepny=\*kolejka;

printf("Podaj liczbe do wstawienia: ");

scanf("%d", &nowy->wartosc);

\*kolejka=nowy;

printf("\nPomyslenie dodano element na poczatek kolejki\nKliknij aby kontynuowac.");

getch();

}

Pop(lifo \*\*kolejka) {

if(IsEmpty(\*kolejka))

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

else {

lifo \*temp=\*kolejka;

lifo \*pop=temp;

temp=temp->nastepny;

free(pop);

\*kolejka=temp;

printf("Pomyslnie usunieto element na poczatku kolejki");

}

printf("\nKliknij aby kontynuowac.");

getch();

}

Clear(lifo \*\*kolejka) {

lifo \*temp=\*kolejka;

while(!IsEmpty(temp)) {

lifo \*pop=temp;

temp=temp->nastepny;

free(pop);

}

\*kolejka=NULL;

printf("Pomyslnie wyczyszczono kolejke\nKliknij aby kontynuowac.");

getch();

}

main() {

lifo \*kolejka=NULL;

int polecenie;

do {

gui();

scanf("%d", &polecenie);

system("cls");

switch(polecenie) {

case 0:

printf("Zakonczono dzialanie programu.");

break;

case 1:

Top(kolejka);

break;

case 2:

Push(&kolejka);

break;

case 3:

Pop(&kolejka);

break;

case 4:

if(IsEmpty(kolejka))

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

else

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

printf("\nKliknij aby kontynuowac.");

getch();

break;

case 5:

Clear(&kolejka);

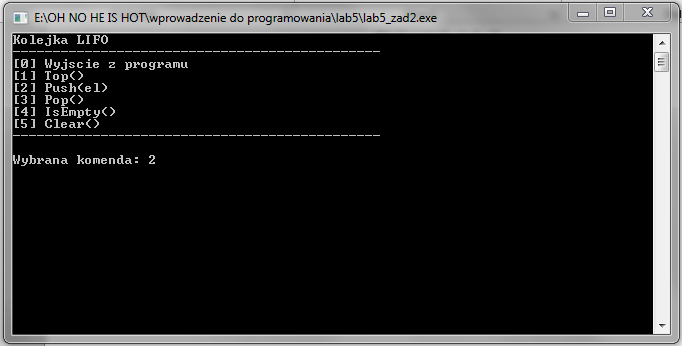
break;

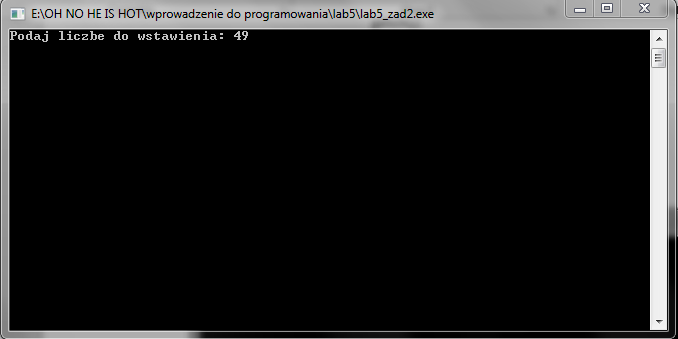
}

} while(polecenie);

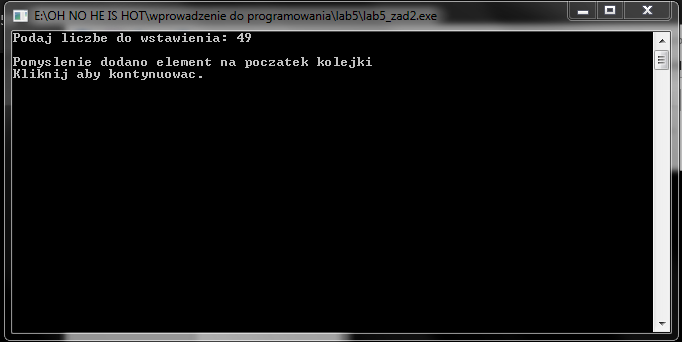
}

* 1. Dane wejściowe





* 1. Dane wyjściowe



1. Wnioski

O(n) – złożoność obliczeniowa zależy liniowo od liczby wprowadzonych elementów.