**STOS LIFO**

#include <stdio.h>

#include <stdlib.h>

struct el\_LIFO

{

int x;

struct el\_LIFO\* nast;

};

//dodanie el na stos

struct el\_LIFO\* push(struct el\_LIFO \*top, int el)

{

struct el\_LIFO \* new\_el\_LIFO=(struct el\_LIFO\*)malloc(sizeof(struct el\_LIFO));

new\_el\_LIFO->x=el;

new\_el\_LIFO->nast=top;

return new\_el\_LIFO;

}

//spr wierzcholka

int top\_el(struct el\_LIFO \*top, int \*el)

{

if (top!=NULL)

{

\*el=top->x;

return 1;

}

else

{

return 0;

}

}

//usuwanie wierzcholka

struct el\_LIFO\* pop(struct el\_LIFO \*top, int \*el)

{

struct el\_LIFO \* temp;

if(top!=NULL)

{

\*el=top->x;

temp=top->nast;

free(top);

return temp;

}

else

{

return NULL;

}

}

//wydruk

struct el\_LIFO\* wydruk(struct el\_LIFO \*top)

{

struct el\_LIFO\* temp=NULL;

int d;

if(top!=NULL)

{

printf("\nGora stosu");

while(top!=NULL)

{

top=pop(top,&d);

temp=push(temp,d);

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

}

while(temp!=NULL)

{

temp=pop(temp,&d);

top=push(top,d);

}

printf("\nDol stosu");

}

else

{

printf("\nStos pusty.");

}

return top;

}

int main()

{

int d, t, x, wybor;

struct el\_LIFO \*top\_LIFO=NULL;

do

{

system("cls");

printf("MENU:");

printf("\n1. Tworz nowy stos.");

printf("\n2. Poloz element na stos.");

printf("\n3. Zdejmij element ze stosu.");

printf("\n4. Pokaz wierzcholek stosu.");

printf("\n5. Wyswietl stos.");

printf("\n6. Koniec programu.");

printf("\nTwoj wybor: ");

scanf("%d", &wybor);

switch(wybor)

{

case 1:

while(top\_LIFO!=NULL)

{

top\_LIFO=pop(top\_LIFO, &d);

}

break;

case 2:

printf("\nPodaj nowy element: ");

scanf("%d", &x);

top\_LIFO=push(top\_LIFO, x);

break;

case 3:

if(top\_LIFO!=NULL)

{

top\_LIFO=pop(top\_LIFO, &d);

printf("\nUsunieto: %d", d);

}

else

{

printf("\nStos pusty, brak elementow do usuniecia.");

}

fflush(stdin);

getchar();

break;

case 4:

if(top\_el(top\_LIFO, &t))

{

printf("\nWierzcholek: %d", top\_LIFO->x);

}

else

{

printf("\nStos pusty.");

}

fflush(stdin);

getchar();

break;

case 5:

top\_LIFO=wydruk(top\_LIFO);

fflush(stdin);

getchar();

break;

}

}

while(wybor!=6);

return 0;

}

**KOLEJKA FIFO**

#include <stdio.h>

#include <stdlib.h>

struct el\_FIFO

{

int x;

struct el\_FIFO \*nast;

};

struct FIFO

{

struct el\_FIFO \*pocz;

struct el\_FIFO \*koniec;

};

//dodanie el do kolejki

struct FIFO \*attache(struct FIFO \*kolejka, int el)

{

struct el\_FIFO \*nowy\_el=(struct el\_FIFO \*)malloc(sizeof(struct el\_FIFO));

nowy\_el->x=el;

nowy\_el->nast=NULL;

if(kolejka->koniec==NULL)

{

kolejka->pocz=nowy\_el;

kolejka->koniec=nowy\_el;

}

else

{

kolejka->koniec->nast=nowy\_el;

kolejka->koniec=nowy\_el;

}

return kolejka;

}

//usuniecie el - "obsluzenie klienta"

struct FIFO \*detache(struct FIFO \*kolejka, int \*el)

{

struct el\_FIFO \*temp;

\*el=kolejka->pocz->x;

if(kolejka->pocz!=kolejka->koniec)

{

temp=kolejka->pocz->nast;

free(kolejka->pocz);

kolejka->pocz=temp;

}

else

{

free(kolejka->pocz);

kolejka->pocz=NULL;

kolejka->koniec=NULL;

}

return kolejka;

}

//wydruk

struct FIFO \*wydruk(struct FIFO \*kolejka)

{

if(kolejka->pocz!=NULL)

{

int d;

struct FIFO \*kolejka2=(struct FIFO \*)malloc(sizeof(struct FIFO));

kolejka2->pocz=NULL;

kolejka2->koniec=NULL;

printf("\nPoczatek kolejki");

while(kolejka->pocz!=NULL)

{

kolejka=detache(kolejka, &d);

kolejka2=attache(kolejka2, d);

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

}

printf("\nKoniec kolejki");

while(kolejka2->pocz!=NULL)

{

kolejka2=detache(kolejka2, &d);

kolejka=attache(kolejka, d);

}

}

else

{

printf("\nKolejka pusta.");

}

return kolejka;

}

int main()

{

int d, wybor;

struct FIFO \*kolejka=(struct FIFO \*)malloc(sizeof(struct FIFO));

kolejka->pocz=NULL;

kolejka->koniec=NULL;

do

{

system("cls");

printf("MENU:");

printf("\n1. Stworzenie pustej kolejki.");

printf("\n2. Dolaczenie elementu do kolejki.");

printf("\n3. Usuniecie elementu z kolejki.");

printf("\n4. Wydruk wszystkich elementow kolejki.");

printf("\n5. Koniec programu.");

printf("\nTwoj wybor: ");

scanf("%d", &wybor);

switch(wybor)

{

case 1:

while(kolejka->pocz)

{

kolejka=detache(kolejka, &d);

}

break;

case 2:

printf("\nPodaj nowy element: ");

scanf("%d", &d);

kolejka=attache(kolejka, d);

break;

case 3:

if(kolejka->pocz!=NULL)

{

kolejka=detache(kolejka, &d);

printf("\nUsunieto: %d", d);

}

else

{

printf("\nKolejka pusta, brak elementow do usuniecia.");

}

fflush(stdin);

getchar();

break;

case 4:

kolejka=wydruk(kolejka);

fflush(stdin);

getchar();

break;

}

}

while(wybor!=5);

return 0;

}

**LISTY – 1kier, cyk, 2kier**

**JEDNOKIERUNKOWA**

#include <stdio.h>

#include <stdlib.h>

struct lista

{

int data;

struct lista \*next;

};

struct lista \*insert(struct lista \*head, int value, int loc)

{

struct lista \*newNode, \*temp;

newNode=(struct lista \*)malloc(sizeof(struct lista));

if(head==NULL)

{

newNode->data=value;

newNode->next=head;

head=newNode;

}

else

{

newNode->data=value;

newNode->next=NULL;

temp=head;

while(temp->data!=loc)

{

if(temp->next==NULL)

{

temp->next=newNode;

return head;

}

else

{

temp=temp->next;

}

}

newNode->next=temp->next;

temp->next=newNode;

}

return head;

}

struct lista \*usun(struct lista\* head, int delValue)

{

struct lista \*temp=NULL;

struct lista \*curr=head;

if(head->data==delValue)

{

head=head->next;

}

else

{

while(curr)

{

if(curr->data==delValue)

{

temp->next=curr->next;

free(curr);

return head;

}

temp=curr;

curr=curr->next;

}

} return head; }

void display(struct lista \*head)

{

if(head==NULL)

{

printf("Lista jest pusta.");

}

else

{

struct lista \*temp=head;

printf("Elementy listy:");

while(temp->next!=NULL)

{

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

temp=temp->next;

}

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

}

}

int main()

{

struct lista\* head=NULL;

head=insert(head, 12, 1);

head=insert(head, 3, 2);

head=insert(head, 11, 12);

head=insert(head, 2, 3);

head=usun(head, 3);

display(head);

return 0;

}

**JEDNOKIERUNKOWA CYKLICZNA**

#include <stdio.h>

#include <stdlib.h>

struct lista

{

int data;

struct lista \*next;

};

struct lista \*insertAfter(struct lista \*head, int value, int location)

{

struct lista \*newNode;

newNode=(struct lista \*)malloc(sizeof(struct lista));

newNode->data=value;

if(head==NULL)

{

head=newNode;

newNode->next=head;

}

else

{

struct lista \*temp=head;

while(temp->data!=location)

{

if(temp->next==head)

{

temp->next=newNode;

newNode->next=head;

return head;

}

else

{

temp=temp->next;

}

}

newNode->next=temp->next;

temp->next=newNode;

}

return head;

}

struct lista \*deleteSpecific(struct lista \*head, int delValue)

{

if(head==NULL)

{

printf("Lista jest pusta");

}

else

{

struct lista \*temp1=head, \*temp2;

while(temp1->data!=delValue)

{

if(temp1->next==head)

{

printf("\nNie ma na liscie podanego el");

return head;

}

else

{

temp2=temp1;

temp1=temp1->next;

}

}

if(temp1->next==head)

{

temp2->next=head;

free(temp1);

}

else

{

if(temp1==head)

{

temp2=head;

while(temp2->next!=head)

temp2=temp2->next;

head=head->next;

temp2->next=head;

free(temp1);

}

else

{

if(temp1->next==head)

{

temp2->next=head;

}

else

{

temp2->next=temp1->next;

}

free(temp1);

}

}

printf("Usunieto.\n");

}

return head;

}

void display(struct lista \*head)

{

if(head==NULL)

{

printf("Lista jest pusta.");

}

else

{

struct lista \*el=head;

printf("Elementy listy:");

while(el->next!=head)

{

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

el=el->next;

}

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

}

}

int main()

{

struct lista \*head=NULL;

head=insertAfter(head, 1, 1);

head=insertAfter(head, 3, 2);

head=insertAfter(head, 4, 1);

head=deleteSpecific(head, 3);

display(head);

return 0;

}

**DUWKIERUNKOWA**

#include <stdio.h>

#include <stdlib.h>

struct el\_listy

{

int x;

struct el\_listy \*lewy;

struct el\_listy \*prawy;

};

struct lista

{

struct el\_listy \*pocz;

struct el\_listy \*koniec;

};

struct el\_listy \*szukaj2(struct lista\* list);

//dodanie el. na poz. o podanym indeksie

struct lista \*dodanie1(int element, struct lista \*lista\_2k)

{

struct el\_listy \*nowy\_el=(struct el\_listy \*)malloc(sizeof(struct el\_listy));

struct el\_listy \*el=(struct el\_listy \*)malloc(sizeof(struct el\_listy));

el=lista\_2k->pocz;

int i=1, indeks;

printf("Podaj indeks elementu: ");

scanf("%d", &indeks);

while(el&&i<indeks)

{

el=el->prawy;

i++;

}

struct el\_listy \*szukany=(struct el\_listy \*)malloc(sizeof(struct el\_listy));

szukany=el;

if(szukany==NULL)

{

nowy\_el->x=element;

nowy\_el->prawy=NULL;

if(lista\_2k->pocz==NULL)

{

lista\_2k->pocz=nowy\_el;

lista\_2k->koniec=nowy\_el;

nowy\_el->lewy=NULL;

}

else

{

nowy\_el->lewy=lista\_2k->koniec;

lista\_2k->koniec->prawy=nowy\_el;

lista\_2k->koniec=nowy\_el;

}

}

else if(indeks==1)

{

nowy\_el->x=element;

nowy\_el->prawy=NULL;

nowy\_el->prawy=lista\_2k->pocz;

lista\_2k->pocz->lewy=nowy\_el;

lista\_2k->pocz=nowy\_el;

}

else

{

nowy\_el->x=element;

nowy\_el->lewy=szukany->lewy;

nowy\_el->prawy=szukany;

szukany->lewy->prawy=nowy\_el;

szukany->lewy=nowy\_el;

}

return lista\_2k;

}

//dodanie el. za el. o podanej wartosci

struct lista \*dodanie2(int element, struct lista \*lista\_2k)

{

struct el\_listy \*nowy\_el=(struct el\_listy \*)malloc(sizeof(struct el\_listy));

struct el\_listy \*szukany=(struct el\_listy \*)malloc(sizeof(struct el\_listy));

printf("Podaj wartosc szukanego elementu: ");

szukany=szukaj2(lista\_2k);

if(szukany==NULL||szukany==lista\_2k->koniec)

{

nowy\_el->x=element;

nowy\_el->prawy=NULL;

if(lista\_2k->pocz==NULL)

{

lista\_2k->pocz=nowy\_el;

lista\_2k->koniec=nowy\_el;

nowy\_el->lewy=NULL;

}

else

{

nowy\_el->lewy=lista\_2k->koniec;

lista\_2k->koniec->prawy=nowy\_el;

lista\_2k->koniec=nowy\_el;

}

}

else

{

nowy\_el->x=element;

nowy\_el->lewy=szukany;

nowy\_el->prawy=szukany->prawy;

szukany->prawy->lewy=nowy\_el;

szukany->prawy=nowy\_el;

}

return lista\_2k;

}

//usuniecie el. o podanym indeksie

struct lista \*usun1(struct lista \*lista\_2k)

{

struct el\_listy \*el=(struct el\_listy \*)malloc(sizeof(struct el\_listy));

el=lista\_2k->pocz;

int i=1, indeks;

printf("\nPodaj indeks elementu: ");

scanf("%d", &indeks);

while(el&&i<indeks)

{

el=el->prawy;

i++;

}

struct el\_listy \*usuwany=(struct el\_listy \*)malloc(sizeof(struct el\_listy));

usuwany=el;

if(usuwany!=NULL)

{

if(usuwany->lewy)

{

usuwany->lewy->prawy=usuwany->prawy;

}

else

{

lista\_2k->pocz=usuwany->prawy;

}

if(usuwany->prawy)

{

usuwany->prawy->lewy=usuwany->lewy;

}

else

{

lista\_2k->koniec=usuwany->lewy;

}

printf("\nUsunieto element.");

}

else

{

printf("\nPodany indeks nie znajduje sie na liscie.");

}

return lista\_2k;

}

//usuniecie el. o podanej wartosci

struct lista \*usun2(struct lista \*lista\_2k)

{

struct el\_listy \*usuwany=(struct el\_listy \*)malloc(sizeof(struct el\_listy));

printf("\nPodaj wartosc szukanego elementu: ");

usuwany=szukaj2(lista\_2k);

if(usuwany!=NULL)

{

if(usuwany->lewy)

{

usuwany->lewy->prawy=usuwany->prawy;

}

else

{

lista\_2k->pocz=usuwany->prawy;

}

if(usuwany->prawy)

{

usuwany->prawy->lewy=usuwany->lewy;

}

else

{

lista\_2k->koniec=usuwany->lewy; }

printf("\nUsunieto element.");

}

else

{

printf("\nPodany element nie znajduje sie na liscie.");

}

return lista\_2k;

}

int szukaj1(struct lista \*lista\_2k)

{

int szukana, indeks=1;

scanf("%d", &szukana);

if(lista\_2k->pocz==NULL)

{

return 0;

}

else

{

struct el\_listy \*el=(struct el\_listy \*)malloc(sizeof(struct el\_listy));

el=lista\_2k->pocz;

while(1)

{

if(el->x==szukana)

{

break;

}

else if(el->prawy)

{

el=el->prawy;

indeks++;

}

else

{

return 0;

}

}

return indeks;

}

}

struct el\_listy \*szukaj2(struct lista \*lista\_2k)

{

int szukana;

scanf("%d", &szukana);

struct el\_listy \*el=(struct el\_listy \*)malloc(sizeof(struct el\_listy));

el=lista\_2k->pocz;

if(lista\_2k->pocz==NULL)

{

return NULL;

}

else

{

while(1)

{

if(el->x==szukana)

{

break;

}

else if(el->prawy)

{

el=el->prawy;

}

else

{

return NULL;

}

}

return el;

}

}

void wydruk1(struct lista \*lista\_2k)

{

if(lista\_2k->pocz==NULL)

{

printf("\nLista jest pusta.");

}

else

{

struct el\_listy \*el=(struct el\_listy \*)malloc(sizeof(struct el\_listy));

el=lista\_2k->pocz;

printf("\nLista od poczatku:");

while(el)

{

printf(" %d", el->x);

el=el->prawy;

}

}

}

void wydruk2(struct lista \*lista\_2k)

{

if(lista\_2k->pocz==NULL)

{

printf("\nLista jest pusta.");

}

else

{

struct el\_listy \*el=(struct el\_listy \*)malloc(sizeof(struct el\_listy));

el=lista\_2k->koniec;

printf("\nLista od konca:");

while(el)

{

printf(" %d", el->x);

el=el->lewy;

}

}

}

int main()

{

struct lista \*lista\_2k=(struct el\_listy \*)malloc(sizeof(struct el\_listy));

lista\_2k->pocz=NULL;

lista\_2k->koniec=NULL;

int wybor, x, indeks;

do

{

system("cls");

printf("MENU:");

printf("\n1. Dodanie elementu do listy.");

printf("\n2. Usuniecie elementu z listy.");

printf("\n3. Wyszukanie elementu na liscie.");

printf("\n4. Wydruk listy.");

printf("\n5. Koniec programu.");

printf("\nTwoj wybor: ");

scanf("%d", &wybor);

switch(wybor)

{

case 1:

system("cls");

printf("Dodanie elementu do listy:");

printf("\n1. Na podana pozycje.");

printf("\n2. Za podany element.");

printf("\nTwoj wybor: ");

scanf("%d", &wybor);

switch(wybor)

{

case 1:

printf("\nPodaj wartosc elementu: ");

scanf("%d", &x);

lista\_2k=dodanie1(x, lista\_2k);

break;

case 2:

printf("\nPodaj wartosc nowego elementu: ");

scanf("%d", &x);

lista\_2k=dodanie2(x, lista\_2k);

break;

}

break;

case 2:

system("cls");

printf("Usuniecie elementu z listy:");

printf("\n1. Z podanej pozycji.");

printf("\n2. O podanej wartoœci.");

printf("\nTwoj wybor: ");

scanf("%d", &wybor);

switch(wybor)

{

case 1:

lista\_2k=usun1(lista\_2k);

fflush(stdin);getchar();

break;

case 2:

lista\_2k=usun2(lista\_2k);

fflush(stdin);getchar();

break;

}

break;

case 3:

printf("\nPodaj wartosc szukanego elementu: ");

indeks=szukaj1(lista\_2k);

if(indeks==0)

{

printf("\n%d - szukanego elementu nie ma na liscie.", indeks);

}

else

{

printf("\nIndeks pierwszego wystapienia elementu: %d", indeks);

}

fflush(stdin);getchar();

break;

case 4:

system("cls");

printf("Wydruk listy:");

printf("\n1. Od poczatku.");

printf("\n2. Od konca.");

printf("\nTwoj wybor: ");

scanf("%d", &wybor);

switch(wybor)

{

case 1:

wydruk1(lista\_2k);

fflush(stdin);getchar();

break;

case 2:

wydruk2(lista\_2k);

fflush(stdin);getchar();

break;

}

break;

}

}

while(wybor!=5);

return 0;

}

**DUWKIERUNKOWA CYK**

#include <stdio.h>

#include <stdlib.h>

struct Node

{

int data;

struct Node \*next;

struct Node \*prev;

};

struct Node \*insertAfter(struct Node \*start, int value, int loc)

{

struct Node \*new\_node=(struct Node \*)malloc(sizeof(struct Node));

new\_node->data=value;

if(start==NULL)

{

new\_node->next=new\_node->prev=new\_node;

start=new\_node;

return start;

}

else

{

struct Node \*temp=start;

while(temp->data!=loc)

{

if(temp->next==start)

{

struct Node \*last=start->prev;

new\_node->next=start;

start->prev=new\_node;

new\_node->prev=last;

last->next=new\_node;

return start;

}

else

{

temp=temp->next;

}

}

struct Node \*next=temp->next;

temp->next=new\_node;

new\_node->prev=temp;

new\_node->next=next;

next->prev=new\_node;

}

return start;

}

struct Node \*deleteNode(struct Node \*start, int value)

{

if(start==NULL)

return start;

struct Node \*curr=start, \*prev\_1=NULL;

while(curr->data!=value)

{

if(curr->next==start)

{

printf("\nList doesn't have node with value = %d", value);

return start;

}

prev\_1=curr;

curr=curr->next;

}

if(curr->next==start&&prev\_1==NULL)

{

start=NULL;

free(curr);

return start;

}

if(curr==start)

{

prev\_1=start->prev;

start=start->next;

prev\_1->next=start;

start->prev=prev\_1;

free(curr);

}

else if (curr->next == start)

{

prev\_1->next=start;

start->prev=prev\_1;

free(curr);

}

else

{

struct Node \*temp=curr->next;

prev\_1->next=temp;

temp->prev=prev\_1;

free(curr);

}

return start;

}

void display(struct Node\* start)

{

struct Node \*temp = start;

printf("Traversal in forward direction \n");

while (temp->next != start)

{

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

temp = temp->next;

}

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

}

int main()

{

struct Node \*start=NULL;

start=insertAfter(start, 1, 1);

start=insertAfter(start, 3, 2);

start=insertAfter(start, 2, 1);

start=deleteNode(start, 2);

display(start);

return 0;

}