Temat zajęć: Algorytmy sortowania w języku C

Autor: mgr inż. Sławomir Samolej

Zagadnienie 1. (Sortowanie)

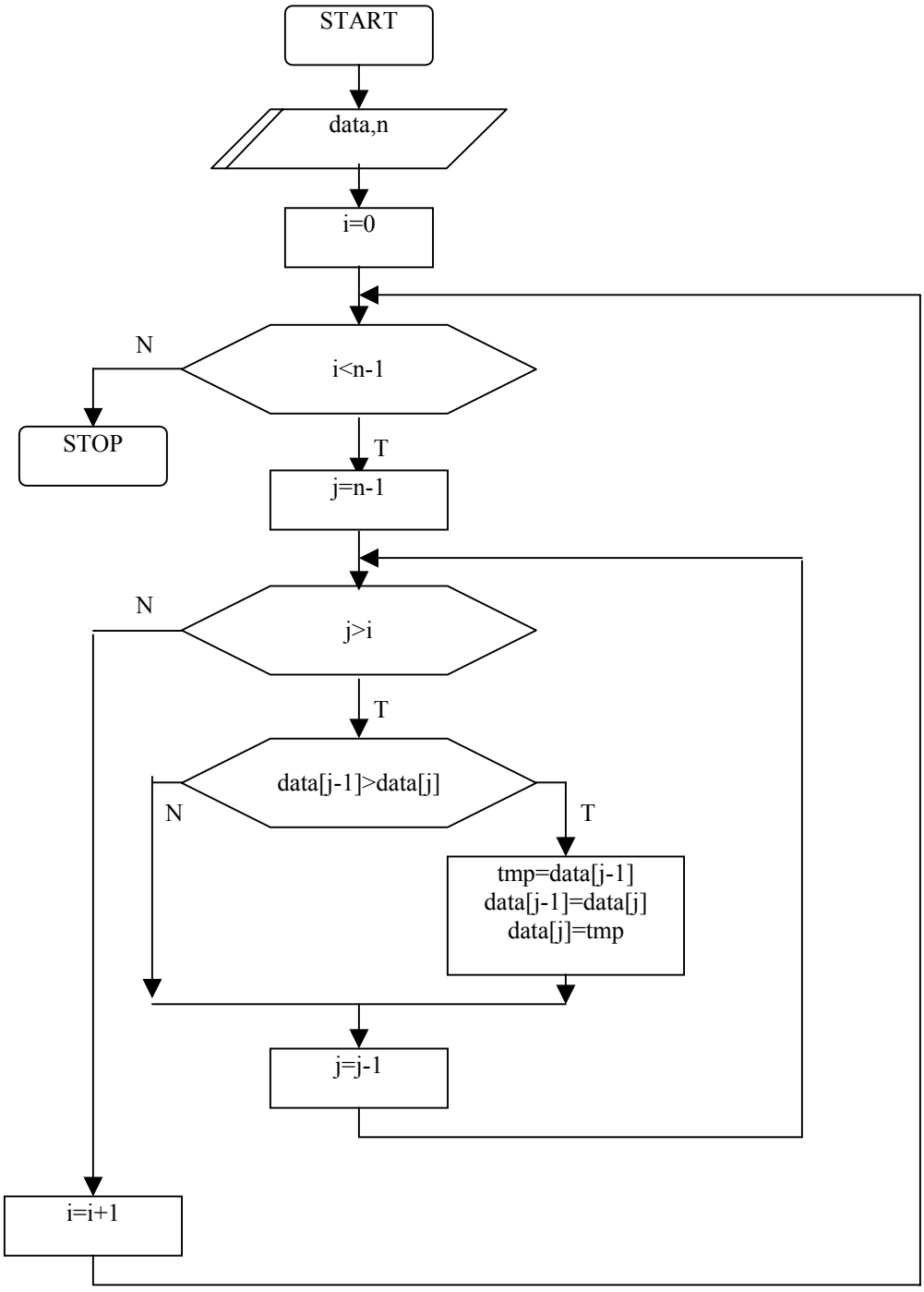
Kolejne zadania dotyczyć będą najprostszych w realizacji algorytmów sortowania. Algorytmy sortowania omówione zostały na wykładzie i ćwiczeniach. Celem obecnych zajęć jest nabycie praktycznej umiejętności implementacji algorytmów w języku C.

Zadanie 1.

Dany jest program:

```
#include <stdio.h>
#define N 7
int dane[N]={0,4,-2,3,5,1,-5};
void sort_babelkowe(int data[], int n);

void main(void)
{
int i;
// wypisanie poczatkowego stanu tablicy
printf("\n");
for(i=0;i<N;i++) printf("%4d",dane[i]);

//sortowanie babelkowe:
sort_babelkowe(dane,N);

//wypisanie stnu tablicy po przesortowaniu
printf("\n");
for(i=0;i<N;i++) printf("%4d",dane[i]);
}

void sort_babelkowe(int data[], int n)
{}
```

Napisać funkcję „sort_babelkowe”, która stosując algorytm sortowania bąbelkowego dokona sortowania tablicy przekazanej do niej przez parametr „data”. Parametr „n” przechwytuje rozmiar tablicy. Poniżej umieszczono przykładowy algorytm sortowania bąbelkowego. Kod funkcji sortującej uzupełnić o wypisywanie stanu tablicy w każdej iteracji sortowania.

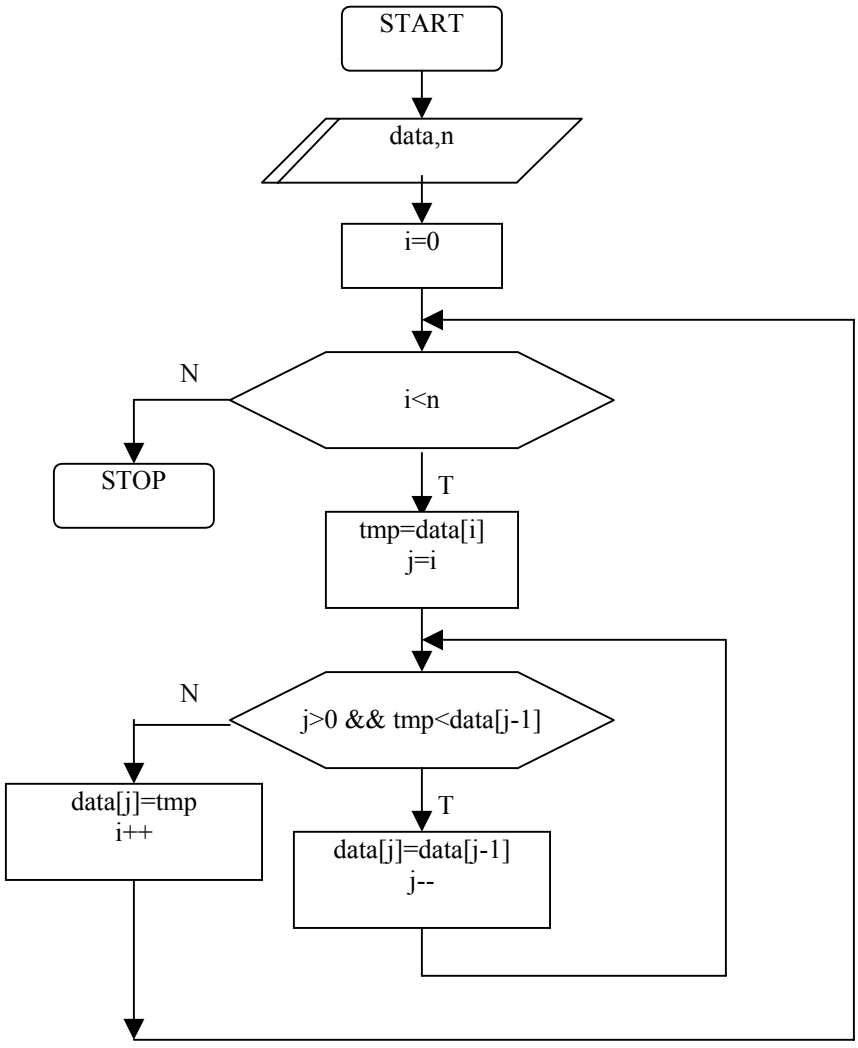

START
data,n
i=0
i<n-1
N
STOP
T
j=n-1
j>i
N
T
data[j-1]>data[j]
N
T
tmp=data[j-1]
data[j-1]=data[j]
data[j]=tmp
j=j-1
i=i+1

Zadanie 2.

Dany jest program:

```
#include <stdio.h>
#define N 7
int dane[N]={0,4,-2,3,5,1,-5};
void sort_przez_wstawienie(int data[], int n);

void main(void)
{
int i;
// wypisanie poczatkowego stanu tablicy
printf("\n");
for(i=0;i<N;i++) printf("%4d",dane[i]);

//sortowanie przez wstawianie:
sort_przez_wstawienie(dane,N);

//wypisanie stnu tablicy po przesortowaniu
printf("\n");
for(i=0;i<N;i++) printf("%4d",dane[i]);
}

void sort_przez_wstawienie(int data[], int n)
{}
```

Napisać funkcję „sort_przez_wstawienie”, która stosując algorytm sortowania przez wstawianie dokona sortowania tablicy przekazanej do niej przez parametr „data”. Parametr „n” przechwytuje rozmiar tablicy. Poniżej umieszczono przykładowy algorytm sortowania przez wstawianie. Kod funkcji sortującej uzupełnić o wypisywanie stanu tablicy w każdej iteracji sortowania.

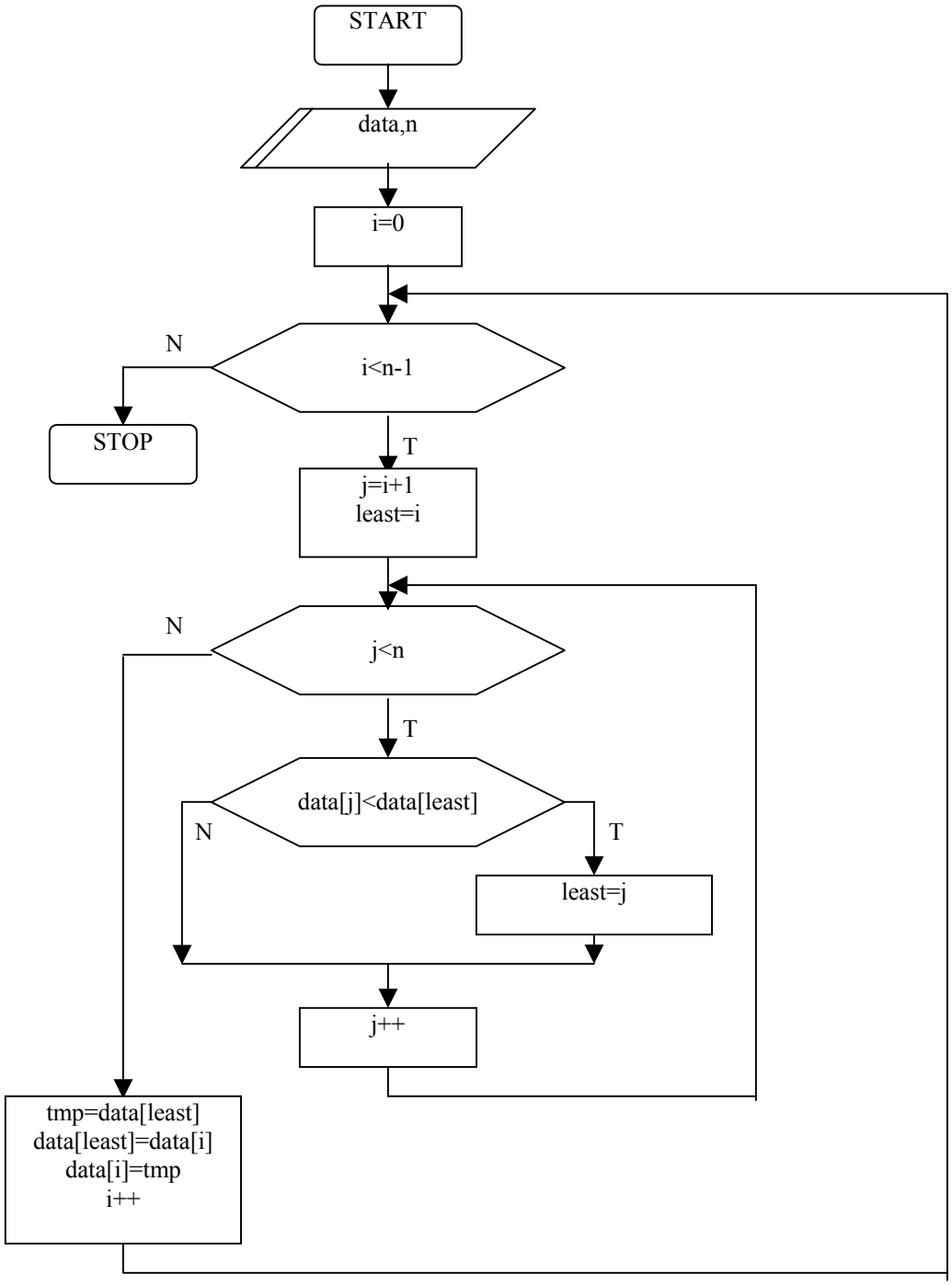

START
data,n
i=0
i<n
N
STOP
T
tmp=data[i]
j=i
j>0 && tmp<data[j-1]
N
data[j]=tmp
i++
T
data[j]=data[j-1]
j--

Zadanie 3.

Dany jest program:

```
#include <stdio.h>
#define N 7
int dane[N]={0,4,-2,3,5,1,-5};
void sort_przez_wybieranie (int data[], int n);

void main(void)
{
int i;
// wypisanie poczatkowego stanu tablicy
printf("\n");
for(i=0;i<N;i++) printf("%4d",dane[i]);

//sortowanie przez wybieranie:
sort_przez_wybieranie (dane,N);

//wypisanie stnu tablicy po przesortowaniu
printf("\n");
for(i=0;i<N;i++) printf("%4d",dane[i]);
}

void sort_przez_wybieranie (int data[], int n)
{}
```

Napisać funkcję „sort_przez_wybieranie”, która stosując algorytm sortowania przez wybieranie dokona sortowania tablicy przekazanej do niej przez parametr „data”. Parametr „n” przechwytuje rozmiar tablicy. Poniżej umieszczono przykładowy algorytm sortowania bąbelkowego. Kod funkcji sortującej uzupełnić o wypisywanie stanu tablicy w każdej iteracji sortowania.


START
data,n
i=0
i<n-1
N
STOP
T
j=i+1
least=i
j<n
N
T
data[j]<data[least]
N
T
least=j
j++
tmp=data[least]
data[least]=data[i]
data[i]=tmp
i++

Zadanie 4. (Zadanie nadobowiązkowe)

Zaproponować implementację dowolnego innego algorytmu sortowania w języku C.

Zagadnienie 2. (Przeszukiwanie binarne)

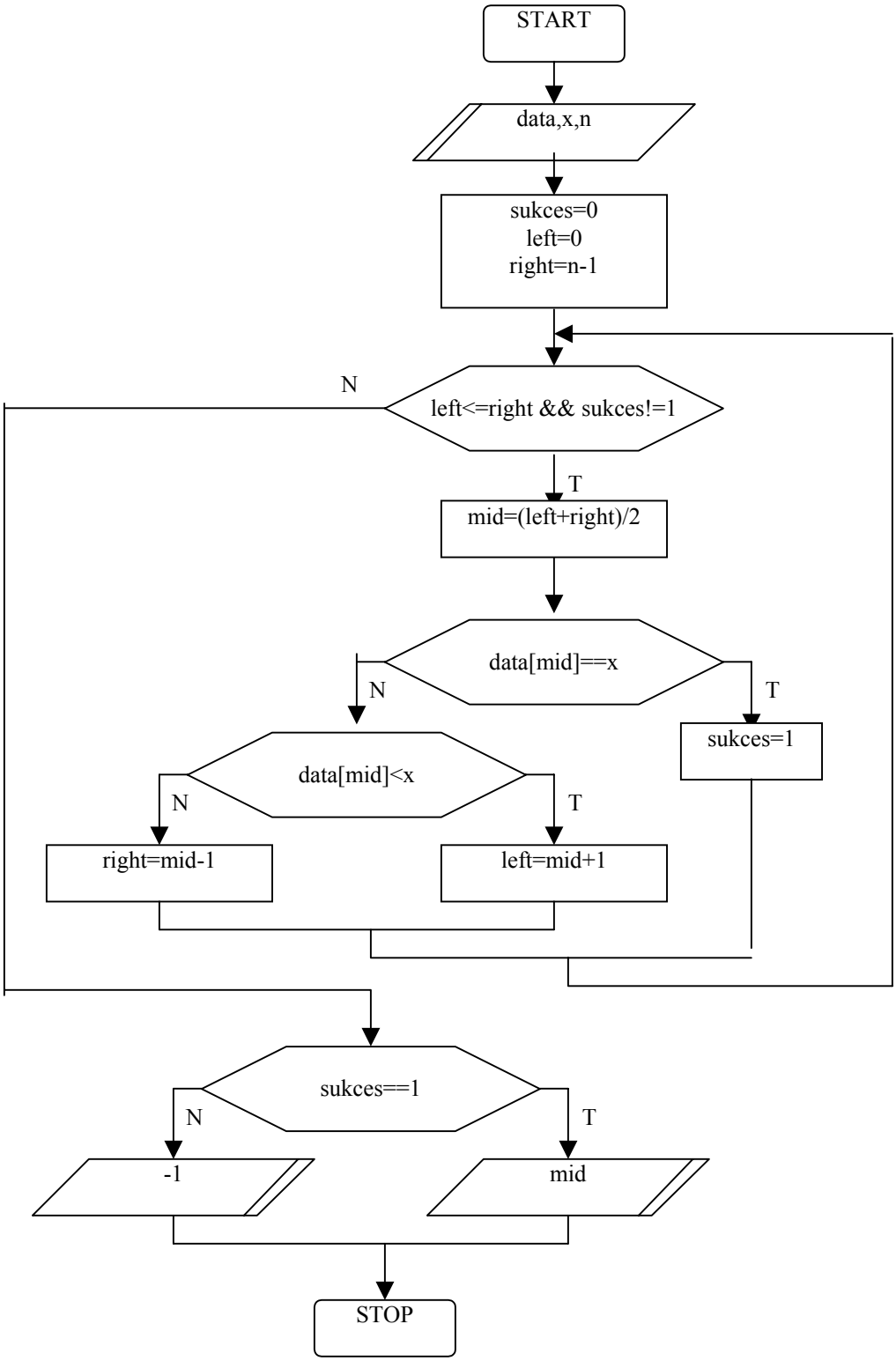
Poniższe zadanie dotyczy implementacji algorytmu przeszukiwania binarnego w języku C wprowadzonego na wykładzie i ćwiczeniach.

Zadanie 1.

Dany jest program:

```
#include <stdio.h>
#define N 7

int dane[N]={-5,-2,0,1,3,4,5};

// przeszukiwanie binarne (wersja iteracyjna)
int szuk_binarne_iter(int data[], int x, int n)
// data - uporzadkowany ciag do przeszukania
// x - element do wyszukania
// n - ilosc elememtow w uporzadkowanym ciagu
{}

void main(void)
{
int i, x;
// wypisanie poczatkowego stanu tablicy
printf("\n");
for(i=0;i<N;i++) printf("%4d",dane[i]);

// wyszukiwanie binarne - iteracyjne
x=4;
i=szuk_binarne_iter(dane,x,N);
printf("\n poszukiwano element:%4d",x);
printf("\n indeks poszukiwanego elementu:%4d",i);
}
```

Zadaniem funkcji „szuk_binarne_iter” jest dokonanie poszukiwania binarnego wartości „x” w ciągu „data” o długości „n”. Funkcja na zwracać pozycję znalezionego elementu w ciągu, lub –1, gdy danej wartości nie odnaleziono. Należy pamiętać, że poszukiwanie binarne może się odbywać tylko w ciągu posortowanym. Poniżej zamieszczono propozycję realizacji algorytmu poszukiwania binarnego.


START
data,x,n
sukces=0
left=0
right=n-1
N
left<=right && sukces!=1
T
mid=(left+right)/2
data[mid]==x
N
T
sukces=1
data[mid]<x
N
T
right=mid-1
left=mid+1
sukces==1
N
T
-1
mid
STOP