Temat zajęć: Zastosowanie wskaźników w języku C

Autor: mgr inż. Sławomir Samolej

**Zagadnienie 1. (Wprowadzenie)**

Wskaźniki to zmienne, które zawierają adres innej zmiennej. Do utworzenia zmiennej wskaźnikowej służy operator „*”, np.:

```
char *wsk1;          // wskaźnik na zmienną typu char
int *ptr1;           // wskaźnik na zmienną typu int
float *f_ptr;        // wskaźnik na zmienną typu float
```

Do przekazania wskaźnikowi adresu pewnej zmiennej służy operator „&”, np.:

```
int* i_ptr;
int a;
a=10;
i_ptr=&a;            //wskaźnikowi i_ptr przypisz adres zmiennej a
```

Aby uzyskać wartość wskazywanej przez wskaźnik zmiennej należy posłużyć się operatorem „*”, np.:

```
char c1='a';
char c2;
char *c_ptr;
c_ptr=&c1;           //wskaźnikowi c_ptr przypisz adres zmiennej c1
c2=*c_ptr;           //do zmiennej c2 wpisz wartość pokazywaną przez wskaźnik c_ptr
```

Podobnie, mając wskaźnik na pewną zmienną można dokonać zmiany zawartości tej zmiennej, np.:

```
float dana1=2.3;
float *f_ptr=&dana1; //wskaźnikowi f_ptr przypisz adres zmiennej dana1
*f_ptr=3.8;          //w miejsce wskazywane przez wskaźnik wpisz 3.8 (oznacza to
                     //, że zmodyfikowano stan zmiennej dana1)
```

**Zadanie 1.**

Znajdujący się poniżej przykładowy program dokonuje podstawowych operacji na wskaźnikach i ilustruje efekty wykonania tych operacji. Należy uruchomić i przeanalizować program.

```
#include <stdio.h>

int a=10,b=0,c;
int *i1_ptr, *i2_ptr;
```

```
void main(void)
{
	printf("Stan zmiennych a, b, c: %d, %d, %d\n",a,b,c);
	i1_ptr=&a;		//wskaznik i1_ptr pokazuje na zmienna a
	printf("Wartosc zmiennej pokazywanej przez wskaznik i1_ptr: %d\n",*i1_ptr);
	b=*i1_ptr;		// zmiennej b przypisuje wartosc, na ktora pokazuje wskaznik i1_ptr
	printf("Stan zmiennych a, b, c: %d, %d, %d\n",a,b,c);
	*i1_ptr=*i1_ptr+1;	// zwiekszam o jeden zawartosc zmiennej pokazywanej przez
				//wskaznik i1_ptr
				// inne sposoby:
				//	*i1_ptr+=1;
				//	++*i1_ptr;
				//	(*i1_ptr)++; // Nawias jest konieczny!
							// Bo inaczej doszłoby do
							// zwiekszenia wartosci wskaznika,
							// a nie zmiennej wskazywanej przez niego
	printf("Stan zmiennych a, b, c: %d, %d, %d\n",a,b,c);
	i2_ptr=i1_ptr;		// jednemu wskaznikowi mozna przypisac inny
					// i wtedy oba pokazuja na
					// to samo miejsce w pamieci
	printf("Wartosc zmiennej pokazywanej przez wskaznik i2_ptr: %d\n",*i2_ptr);
}
```

**Zagadnienie 2. (Wskaźniki jako parametry wywołania funkcji)**

W języku C można przekazywać parametry do funkcji na dwa sposoby: przez wartość i przez wskazanie na zmienną. Gdy parametry są przekazywane przez wartość, to po zakończeniu działania funkcji stan zmiennych, które posłużyły do przekazania informacji o swoim stanie nigdy nie ulega zmianie. Przekazanie parametru przez wskazanie na zmienną (wskaźnik) umożliwia dokonania modyfikacji tej zmiennej wewnątrz funkcji. Wyjaśnieniu tego zagadnienia posłuży przykład znajdujący się poniżej.

## Zadanie 2.

Dany jest program:

```
#include <stdio.h>

int a=4,b=8;
int c=3,d=7;

void swap1(int x, int y);
void swap2(int *x, int *y);

void main(void)
```

```
{
	printf("Zmienne a, b przed wykonaniem funkcji swap1: %d, %d\n",a,b);
	printf("Zmienne c, d przed wykonaniem funkcji swap2: %d, %d\n",c,d);
	swap1(a,b);		// przekazanie parametrow przez wartosc
	swap2(&c,&d);		// przekazanie parametrow przez wskazanie na zmienne
	printf("Zmienne a, b po wykonaniu funkcji swap1: %d, %d\n",a,b);
	printf("Zmienne c, d po wykonaniu funkcji swap2: %d, %d\n",c,d);
}

void swap1(int x, int y)
{
	int tmp;
	tmp=x;
	x=y;
	y=tmp;
}

void swap2(int *x, int *y)
{
	int tmp;
	tmp=*x;
	*x=*y;
	*y=tmp;
}
```

Należy porównać i wyjaśnić wyniki działania funkcji swap1 i swap2.

**Zagadnienie 3. (Wskaźniki i tablice jednowymiarowe)**

Jeśli wskaźnik pokazuje na pewien element tablicy, to zwiększenie wskaźnika o 1 spowoduje, że będzie on pokazywał na kolejny element tablicy, np.:

```
int tab[5]={1,3,4,5,6};
int *t_ptr;

t_ptr=&tab[2]; // wskaznikowi t_ptr przypisano adres 3 elementu tablicy
t_ptr=t_ptr+1; // wskaznik zwiekszono o jeden i teraz pokazuje na 4 element tablicy
```

Żeby wskaźnik wskazywał poprzedni element tablicy, należy wskaźnik zmniejszyć o jeden.

Nazwa tablicy jest wskaźnikiem na jej początek! Przykładowo:

```
int tab[5]={1,3,4,5,6};
int *t1_ptr, *t2_ptr;
t1_ptr=&tab[0];
t2_ptr=tab;
```

Oba wskaźniki t1_ptr i t2_ptr pokazują na to samo miejsce w pamięci.

**Zadanie 3.**

Uruchomić i przeanalizować wyniki działania przykładowego programu:

```
#include <stdio.h>

int tab[5]={23,5,66,-3,5};

void main(void)
{
	int i;
	int *t1_ptr;
	t1_ptr=&tab[0];
	for(i=0;i<5;i++)
	{
		printf("%4d",*t1_ptr);
		t1_ptr++;
	}
	putchar('\n');

	t1_ptr=tab;
	for(i=0;i<5;i++)
	{
		printf("%4d",*t1_ptr);
		t1_ptr++;
	}
	putchar('\n');
}
```

**Zadanie 4.**

Dany jest program:

```
#include <stdio.h>

char txt1[]="To jest tekst";
char txt2[50];

void str_cpy(char* src, char* dst);

void main(void)
{
	printf("Tablica zrodlowa:\n%s\n",txt1);
```

```
	printf("Tablice docelowa:\n%s\n",txt2);
	str_cpy(txt1,txt2);
	printf("Tablica zrodlowa:\n%s\n",txt1);
	printf("Tablice docelowa:\n%s\n",txt2);
}

void str_cpy(char* src, char* dst)
{
	while(*src!='\0')
	{
		*dst=*src;
		dst++;
		src++;
	}
	*dst='\0';
}
```

Program stosuje wskaźnikową wersję funkcji str_cpy do przekopiowania łańcucha znaków z tablicy txt1 do txt2 i może być wskazówką w zadaniu. Zadanie polega na uzupełnieniu następującego programu:

```
#include <stdio.h>

char txt1[100]="To jest tekst";
char txt2[100]="To jest drugi tekst";

void str_cat(char* src, char* dst);

void main(void)
{
	printf("Tablica txt1:\n%s\n",txt1);
	printf("Tablice txt2:\n%s\n",txt2);
	str_cat(txt1,txt2);
	printf("Tablica txt1:\n%s\n",txt1);
	printf("Tablice txt2:\n%s\n",txt2);
}

void str_cat(char* src, char* dst)
{
}
```

Funkcja str_cat ma dołączyć tekst src na koniec tekstu dst.

**Zadanie 5.**

Dany jest program:

```
#include <stdio.h>

char tekst[100];

int rindex(char *s,char t);

void main(void)
{
        char wzorzec; int index;
        printf("Podaj tekst (do 80 znakow:\n");
        gets(tekst);
        printf("Podaj poszukiwany wzorzec (literę):\n");
        scanf("%c",&wzorzec);
        index=rindex(tekst,wzorzec);
        printf("\nOstatnie wystapienie wzorca: %c\nw tekscie: %s\n",wzorzec,tekst);
        if(index!=-1)   printf("wystapiło na %d pozycji tekstu\n",index);
        else                    printf("nie wystapilo\n\n");
}

int rindex(char *s, char wz)
{
}
```

Napisać funkcję „rindex” zwracającą pozycję pierwszego od końca wystąpienia danej litery (wzorca „wz”) w tekście wskazywanym przez „s”. Jeśli wzorzec nie występuje w tekście, to funkcja zwraca wartość –1. Przykładowo wykonanie funkcji:

```
char tekst[]=”To jest tekst”
char c=’t’;
int rez;

rez=rindex(tekst,c);
```

powinno spowodować, że zmienna rez będzie wynosić 12, ponieważ ostatnie wystąpienie litery ‘t’ w tekście „To jest tekst” następuje na 12 pozycji tablicy tekstowej (indeksowanie zaczyna się od 0).

**Zadanie 6.**

Dany jest program:

```
#include <stdio.h>
#include <string.h>

char tekst[100];

void reverse(char *s);

void main(void)
{
	printf("Podaj tekst (do 80 znakow:\n");
	gets(tekst);
	reverse(tekst);
	printf("Odwrocony tekst:\n");
	puts(tekst);
}

void reverse(char *s)
{
}
```

Napisać funkcję „reverse”, która odwraca tekst s. Odwrócenie tekstu polega na zamianie pierwszego elementu z ostatnim, drugiego z przedostatnim i tak dalej. Przykładowo tekst: „To jest tekst” powinien zostać zamieniony na „tsket tsej oT”.