Temat zajęć: **Funkcje w języku C**

Autor: mgr inż. Sławomir Samolej

Zagadnienie 1. (instrukcja wyboru: switch)

Instrukcji „switch” używa się wtedy, gdy zachodzi potrzeba dokonania wyboru na podstawie pewnego wyrażenia liczbowego o ograniczonym zbiorze wartości.
Postać instrukcji „switch” jest następująca:

```
switch(wyrażenie)
{
        case wyrażenie stałe 1:
                                        lista instrukcji 1
        case wyrażenie stałe 2:
                                        lista instrukcji 2
        ...
        case wyrażenie stałe 3:
                                        lista instrukcji n
        default:
                                        awaryjna lista instrukcji
}
```

Przykład:

```
        #include <stdio.h>
        void main(void)
        {
        int n;
        scanf("%d",&n);
        switch(n)
        {
                case 0  : printf("Zero\n");
                case 1  :
                case 2  : printf("Mała\n");
                                break;
                case 3  :
                case 4  :
                case 5  : printf("Średnia\n");
                default : printf("Duża\n");
        }
        }
```

W czasie wykonywania instrukcji „switch” w pierwszej kolejności następuje obliczenie wyrażenia znajdującego się w nawiasie po słowie kluczowym „switch”. Wynik wyrażenia porównywany jest z zestawem stałych wyrażeń podawanych po słowach kluczowych „case”, a

zakończonych znakiem „:”. Jeśli wartość wyrażenia w nawiasie po słowie „switch” jest identyczna z wartością pewnego wyrażenia stałego, następuje wykonanie wszystkich instrukcji znajdujących się po tym wyrażeniu stałym. W instrukcji „switch” zamiast słowa kluczowego „case” można w jednym miejscu wstawić słowo „default”. Jeśli w czasie wykonywania instrukcji „switch” żadne z wyrażeń stałych po słowach „case” nie będzie odpowiadać wyrażeniu po słowie kluczowym „switch”, to w czasie wykonywania instrukcji „switch” wykonane zostaną instrukcje umieszczone po słowie kluczowym „default”. Częstą praktyką jest stosowanie wewnątrz instrukcji „switch” instrukcji „break”. Umożliwia to przerwanie ciągu wykonywanych instrukcji.

Zadania:

- Dany jest przykładowy program, uruchomić i wyjaśnić wynik jego działania, dla następujących wartości wejściowych: 0, 1, 4, 10, -5.
  Program przykładowy:

```
#include <stdio.h>
    void main(void)
    {
    int n;
    scanf("%d",&n);
    switch(n)
    {
            case 0  : printf("Zero\n");
            case 1  :
            case 2  : printf("Mała\n");
                              break;
            case 3  :
            case 4  :
            case 5  : printf("Średnia\n");
            default : printf("Duża\n");
    }
    }
```

- Dany jest przykładowy program, uruchomić i wyjaśnić wynik jego działania, dla następujących wartości wejściowych: 0, 1, 4, 10, -5.
  Program przykładowy:

```
#include <stdio.h>
    void main(void)
    {
    int n;
    scanf("%d",&n);
    switch(n)
    {
            case 0  : printf("Zero\n"); break;
            case 1  :
```

```
        case 2  : printf("Mała\n");  break;
        case 3  :
        case 4  :
        case 5  : printf("Średnia\n"); break;
        default : printf("Duża\n"); break;
    }
    }
```

- Dany jest przykładowy program, uruchomić i wyjaśnić wynik jego działania, dla następujących wartości wejściowych: 0, 1, 4, 10, -5.
  Program przykładowy:

```
#include <stdio.h>
    void main(void)
    {
    int n;
    scanf("%d",&n);
    switch(n)
    {
        case 0  : printf("Zero\n");
        case 1  :
        case 2  : printf("Mała\n");
                    break;
        default : printf("Duża\n");
        case 3  :
        case 4  :
        case 5  : printf("Średnia\n");
    }
    }
```

- Dany jest przykładowy program, uruchomić i wyjaśnić wynik jego działania, dla następujących wartości wejściowych: 0, 1, 4, 10, -5.
  Program przykładowy:

```
#include <stdio.h>
    void main(void)
    {
    int n;
    scanf("%d",&n);
    switch(n)
    {
        case 0  : printf("Zero\n");
        case 1  :
        case 2  : printf("Mała\n");  break;
        default : printf("Duża\n"); break;

        case 3  :
```

```
            case 4  :
            case 5  : printf("Średnia\n");
    }
    }
```

- Dany jest program wyliczający ilość spacji i kropek w linii tekstu z zastosowaniem instrukcji „switch”. Należy uzupełnić program w taki sposób, aby obliczał ilość cyfr i tabulacji.
  Dany program:

```
#include <stdio.h>
void main(void)
{
int c, n_space, n_dot;
c=n_space=n_dot=0;
do
{
    c=getchar();
    switch(c)
    {
            case '.'        :       n_dot++;
                                            break;

            case ' '        :       n_space++;
                                    break;
            default         :       break;
    }
}
while(c!='\n');
printf("Kropki: %d, Spacje: %d",n_dot,n_space);
}
```

Zagadnienie 2. (definiowanie funkcji w języku C)

Pojęcie funkcji wprowadzono w języku C w celu umożliwienia tworzenia podprogramów – fragmentów programów, które mają zdefiniowany interfejs z otoczeniem i mogą być wykorzystywane wielokrotnie w obrębie danego programu lub stosowane w wielu pisanych programach. Typowy program w języku C jest zestawem definicji funkcji oraz sposobu ich wywoływania.
Przykład:

```
#include <stdio.h>

int suma(int a, int b);  //deklaracja funkcji

    void main(void)
    {
        int x,y,s;
        x=6;
        y=8;
        printf("x=%d\ty=%d\n",x,y);
        s=suma(x,y);
        printf("suma=%d\n",s);
        s=suma(3,4);
        printf("suma=%d\n",s);
    }

    int suma(int a, int b)        //definicja funkcji
    {
        int c;
        c=a+b;
        return c;
    }
```

Do funkcji przekazuje się dane poprzez jej parametry (listę nazw i typów danych umieszczoną w nawiasie po nazwie funkcji). W czasie definiowania funkcji określa się również typ danych, jaki funkcja może zwracać (typ danych podany przed nazwą funkcji). Funkcja w programie rozpoznawana jest przez swoją nazwę. Niezależnie od położenia i ilości zadeklarowanych w programie funkcji, program zawsze rozpoczyna swoje działanie od wywołania funkcji main. Przesłanie danych do funkcji odbywa się przez wypełnienie listy jej parametrów, np.:

```
suma(1,4)
```

Przejęcie wyniku zwracanego przez funkcję odbywa się przy pomocy operatora przypisania:

```
s=suma(23,44)
```

Zadania:

- Dany jest przykładowy program:

```
#include <stdio.h>

float fabs(float x);          //deklaracja funkcji

void main(void)
{
float a, w_bz;
printf("Podaj liczbę zmiennopozycyjna:");
scanf("%f",&a);
w_bz=fabs(a);
printf("|a|=%f",w_bz);
}

float fabs(float x) //definicja funkcji
{
}
```

Uzupełnić treść funkcji fabs, w taki sposób, aby wyliczała ona wartość bezwzględną z wprowadzanej do niej liczny zmiennopozycyjnej i zwracała ją.

- Dany jest przykładowy program:

```
#include <stdio.h>

int ile_zerowych_bitow(unsigned char a);        //deklaracja funkcji

void main(void)
{
    unsigned char dana=0x11;
    int wyn;
    wyn=ile_zerowych_bitow(dana);
    printf("liczba: %x (szesnastkowo) zawiera: %d zerowych bitow\n",dana,wyn);
}

int ile_zerowych_bitow(unsigned char a)         //definicja funkcji
{
}
```

Zaproponować treść funkcji “ile_zerowych _bitów”, która obliczać będzie ilość bitów będących zerami w danej przekazywanej do funkcji i która będzie zwracała obliczoną wartość.

- Dany jest przykładowy program:

```
#include <stdio.h>

int czy_duza(char a);       //deklaracja funkcji
```

```
void main(void)
{
    char c;
    int l_duza=0;
    do
    {
            c=getchar();
            if(czy_duza(c)==1) l_duza++;
    }while(c!='\n');
    printf("W linii tekstu było %d duzych liter",l_duza);
}

int czy_duza(char a)        //definicja funkcji
{
}
```

Zaproponować treść funkcji “czy_duza”, która ma zwracać wartość 1, gdy jej parametr wywołania jest dużą literą lub 0 w przeciwnym wypadku. Przykładowy program, który podano powyżej stosuje funkcję „czy_duza” do wyliczenia ilości dużych liter we wprowadzonej linii tekstu.

- Dany jest przykładowy program:

```
#include <stdio.h>

char na_duza(char a);      //deklaracja funkcji

void main(void)
{
    char c;
    do
    {
            c=getchar();
            c=na_duza(c);
            putchar(c);
    }while(c!='\n');
}

char na_duza(char a)       //definicja funkcji
{
}
```

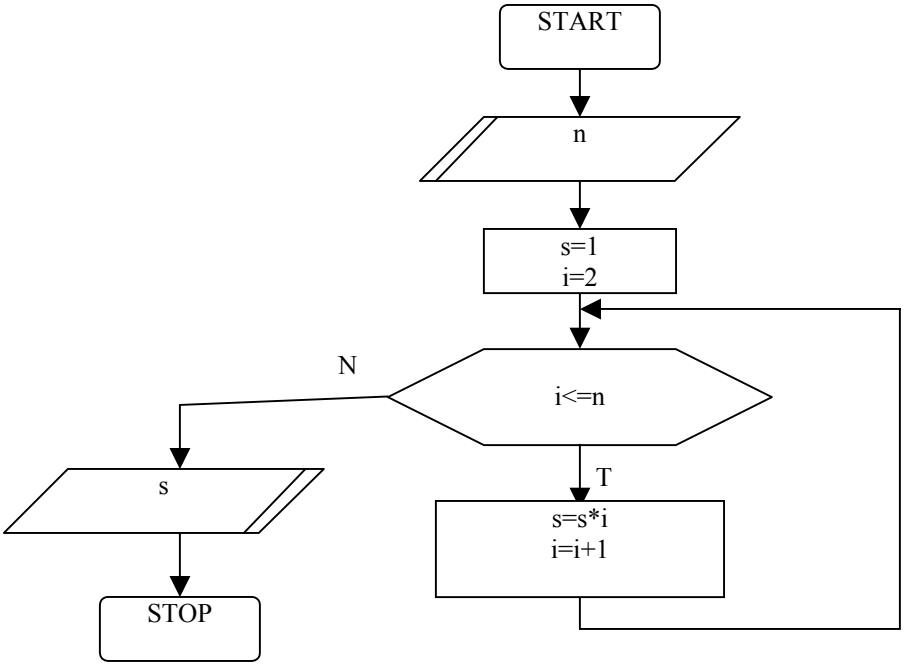
Zaproponować treść funkcji “na_duza”, która ma zamieniać wszystkie małe litery na duże, pozostałe zaś znaki pozostawiać bez zmian. Przykładowy program, który podano powyżej

stosuje funkcję „na_duza” do przekształcenia linii tekstu w taki sposób, aby wszystkie małe litery tekstu zapisane były jako duże.

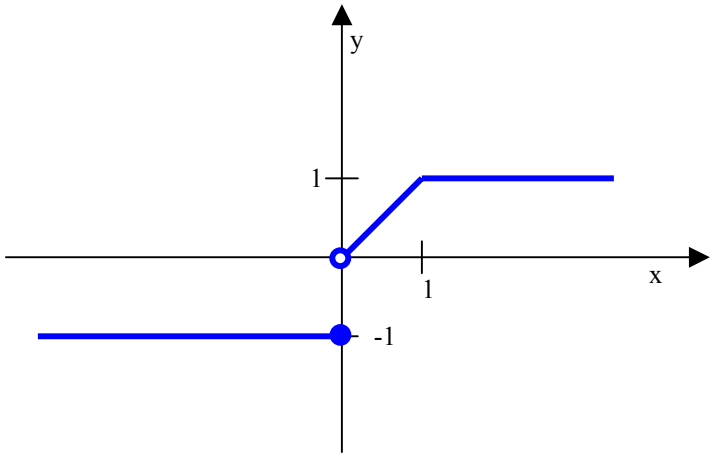
- Dany jest program:

```
#include <stdio.h>

long silnia(long n);          //deklaracja funkcji

void main(void)
{
    long n,s;
    printf("Podaj, dla jakiej liczby ma zostać obliczona silnia:");
    scanf("%ld",&n);
    s=silnia(n);
    printf("%ld ! = %ld",n,s);
}

long silnia(long n)           //definicja funkcji
{
}
```

Należy uzupełnić funkcję silnia w taki sposób, aby obliczała i zwracała silnię z parametru n. Poniżej zamieszczono algorytm obliczania silni:



- Dany jest wykres funkcji:



Dany jest program:

```
#include <stdio.h>

float f1(float x);          //deklaracja funkcji

void main(void)
{
    float a=-3.0, b=0.45, c=11.0, f1a, f1b, f1c;
    f1a=f1(a);
    f1b=f1(b);
    f1c=f1(c);
    printf("\n f1(%f) = %f",a, f1a);
    printf("\n f1(%f) = %f",b, f1b);
    printf("\n f1(%f) = %f",c, f1c);
}

float f1(float x)
{

}
```

Należy uzupełnić funkcję f1 w taki sposób, aby obliczała i zwracała wartości zgodne z danym wykresem funkcji.