1. Rekurencja.

Zadanie 1 (C++) Mamy do rozwiązania następujący problem:

- Dysponujemy tablicą n liczb całkowitych tab[n]={tab[0],tab[1],…,tab[n-1]},
- Czy w tablicy tab występuje liczba x (podana jako parametr)?

[p1.cpp]

```
#include <iostream>
#include <cstdlib>

using namespace std;
const int n=10;
int tab[n]={12,3,2,-7,44,5,1,0,-3};

void szukaj(int tab[n], int left, int right, int x){

    if(left>right){
        cout<<"Element "<<x<<" nie zostal znaleziony\n";
    }
    else{
        if(tab[left]==x){
            cout<<"Znalazlem szukany element "<<x<<endl;
        }
        else
            szukaj(tab,left+1,right,x);
    }
}

int main(){

szukaj(tab,0,n-1,7);
szukaj(tab,0,n-1,5);

system("pause");
return 0;

}
```

Zadanie 2 (C++) Napisać prostym program, który zajmie się obliczeniem silni w sposób rekurencyjny.

[p2.cpp]

```
#include <iostream>
#include <cstdlib>

using namespace std;

unsigned long int silnia(int x){
if(x==0)
    return 1;
else
    return x*silnia(x-1);
}
```

```
//rekurencja z parametrem pomocniczym
unsigned long int silnia2(int x, int temp=1){

if(x==0)
    return temp;
else
    return silnia2(x-1,x*temp);

}

int main(){

cout << "silnia(5)="<<silnia(5)<<endl;
cout << "silnia2(5)="<<silnia2(5)<<endl;

system("pause");
return 0;

}
```

Zadanie 3 (C++) Napisz program wyznaczający elementy tzw. ciągu Fibonacciego. Występuje on często w przyrodzie i jest definiowany następująco:

fib(0)=0;
fib(1)=1,
fib(n)=fib(n-1) + fib(n-2), gdzie n>=2

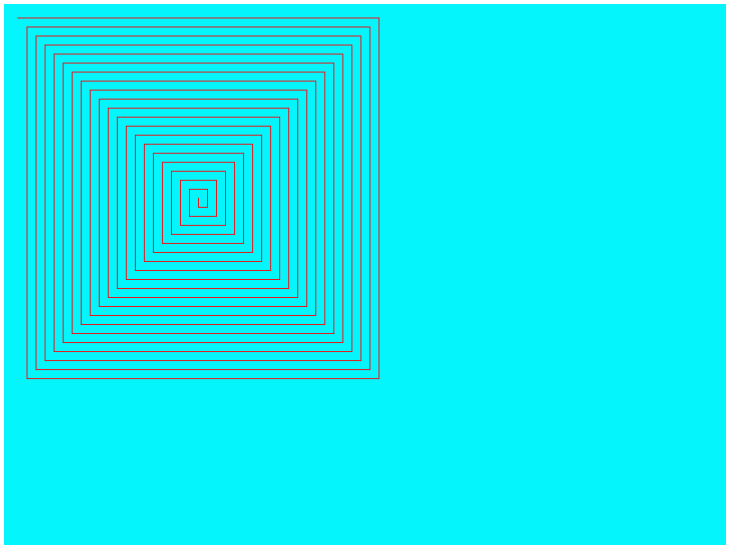
Elementy tego ciągu stanowią liczby naturalne tworzące o takiej własności, że każdy kolejny wyraz (z wyjątkiem dwóch pierwszych) jest sumą dwóch poprzednich (tj. 1, 1, 2, 3, 5, 8, 13, …).

[p3.cpp]

```
#include <iostream>
#include <cstdlib>
//Ciag Fibonacciego
using namespace std;

unsigned long int fib(int x){

    if(x<2){
        return x;
    }
    else{
        return fib(x-1)+fib(x-2);
    }
}

int main(){

cout << "fib(6)="<<fib(6)<<endl;

system("pause");
return 0;

}
```

Zadanie 3 a) Narysuj drzewo wywołań rekurencyjnych funkcji fib(4) z zadania **Z3**.

Zadanie 4 (C++) Napisz program rysujący spiralę (rys. 1) rekurencyjnie jednym pociągnięciem kreski.



Rysunek 1.

Utworzyć nowy projekt (allegro static):
Nowy project -> MultiMedia -> Allegro application (static) [Project C++].

[main.cpp]

```
#include <allegro.h>
#define m 800
#define n 600

void init();
void deinit();
void spirala(int x, int y, int xl, int yl, int dx, int dy, int k);
char a[m+2][n+2], b[m+1][n+1];

BITMAP *bufor=NULL;

int main(){
      allegro_init();
      install_keyboard();
      set_color_depth(24);
      set_gfx_mode(GFX_AUTODETECT_WINDOWED,m,n,0,0);
      set_palette(default_palette);
      clear_to_color(screen,makecol24(0,244,255));
      bufor=create_bitmap(m,n);
    clear_to_color(bufor,makecol24(0,244,255));

    spirala(15,15,400,400,10,10,20);
    //blit(bufor,screen,0,0,0,0,m,n);

    while(!key[KEY_ESC]){
        readkey();
        if(key[KEY_F9]){save_bitmap("spirala.bmp", bufor,
default_palette);};
```

```
    }

    destroy_bitmap(bufor);
    deinit();
    return 0;
}
END_OF_MAIN()

void init() {
      int depth, res;
      allegro_init();
      depth = desktop_color_depth();
      if (depth == 0) depth = 32;
      set_color_depth(depth);
      res = set_gfx_mode(GFX_AUTODETECT_WINDOWED, 640, 480, 0, 0);
      if (res != 0) {
            allegro_message(allegro_error);
            exit(-1);
      }

      install_timer();
      install_keyboard();
      install_mouse();
      /* add other initializations here */
}

void deinit() {
      clear_keybuf();
      allegro_exit();
}

void spirala(int x, int y, int xl, int yl , int dx, int dy, int k){
    int c=0xff0000;
    if(k>0){
        hline(screen,x,y,x+xl,c);//x1,y1,x2
        rest(50);
        vline(screen,x+xl,y,y+yl,c);//x1,y1,y2
        rest(50);
        hline(screen,x+dx,y+yl,x+xl,c);
        rest(50);
        vline(screen,x+dx,y+dy,y+yl,c);
        rest(50);
        x=x+dx;y=y+dy;xl=xl-2*dx;yl=yl-2*dy;
        spirala(x,y,xl,yl,dx,dy,k-1);
    }
    /*
    program mozna wykonac z buforowaniem:

    zakomentowac rest(50),
    zastapic screen na bufor,
    odkomentowac (wewnatrz main()): blit(bufor,screen,0,0,0,0,m,n);
    */
    /*
    zamiast if(k>0) mozna uzyc:
    if(xl>0&&yl>0)
    */
}
```

Zadanie 5 (C++) Napisz program odwracający w sposób rekurencyjny tablicę liczb całkowitych.

[p5.cpp]

```
#include <iostream>
#include <cstdlib>

using namespace std;
int t[10]={0,1,2,3,4,5,6,7,8,9};

void reverse(int * m, int left, int right){
    if(right>left){
        int temp=m[left];
        m[left]=m[right];
        m[right]=temp;
        reverse(m,left+1,right-1);
    }
}

int main(){

cout << "Tablica"<<" t[] przed odwroceniem:\n";
for(int i=0;i<10;i+=1){
cout<<"t["<<i<<"]="<<t[i]<<",";
}

reverse(t,0,9);
cout << "\nTablica"<<" t[] po odwroceniu:\n";
for(int i=0;i<10;i+=1){
cout<<"t["<<i<<"]="<<t[i]<<",";
}
cout<<endl;

system("pause");
return 0;

}
```

Zadanie 6 (C++) Narysuj drzewo wywołań rekurencyjnych funkcji MacCarthy'ego:

```
unsigned long int MacCarthy(int x){
    if(x>100){
        return x-10;
    }
    else{
        return MacCarthy(MacCarthy(x+11));
    }
}
```

dla granicznej liczby x=100 i dla x=99. Następnie sprawdź wyniki pisząc program.

[p6.cpp]

```
#include <iostream>
#include <cstdlib>

using namespace std;
int temp=0;
```

```
unsigned long int MacCarthy(int x){
    temp++;
    if(x>100){
        return x-10;
    }
    else{
        return MacCarthy(MacCarthy(x+11));
    }
}

int main(){

for(int i=90;i<=100;i+=1){
temp=0;
cout << "MacCarthy("<<i<<")="<<MacCarthy(i)<<" ";
cout << temp << endl;
}
system("pause");
return 0;

}
```

Zadanie 7 (C++) Napisz program, który wczytując liczbę całkowitą dodatnią wypisze jej reprezentację w systemie dwójkowym.

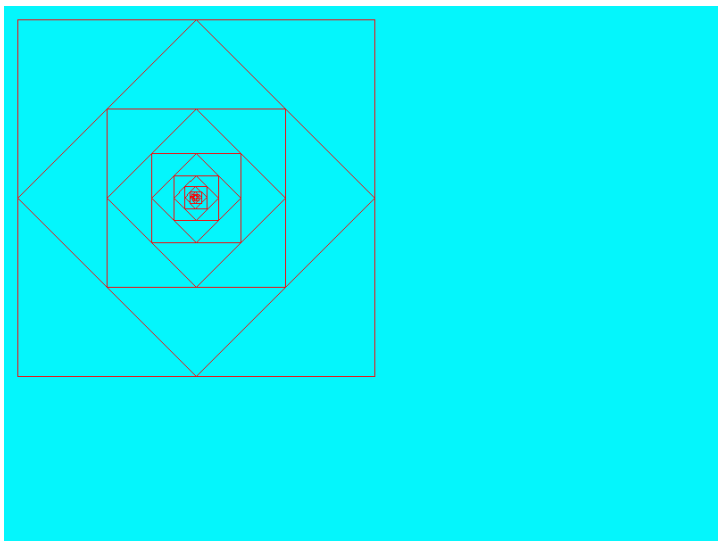
Wskazówka: wykorzystaj znany algorytm dzielenia przez podstawę systemu.
Przykładowo dla liczby 13:
13 : 2 = 6 + 1,
6 : 2 = 3 + 0,
3 : 2 = 1 + 1,
1 : 2 = **0** + 1. (**0** – koniec programu).

[p7.cpp]

```
#include <iostream>
#include <cstdlib>

//reprezentacja dwojkowa

using namespace std;

void binary(int a){
    if(a>0){
        if((a%2)==0){
            a/=2;
            binary(a);
            cout<<0;
        }
        else{
            a/=2;
            binary(a);
            cout<<1;
        }
    }
}
```

```
void binary2(int a){
    if(a!=0){
        binary(a/2);     //a modulo 2
        cout<<a%2;       //reszta z dzielenia przez 2
    }
}

int main(){

int temp;
cout<<"Zamiana na postac dwojkowa\nWpisz liczbe calkowita dodatnia:\n";
cin>>temp;

binary(temp);
cout<<endl;

binary2(temp);
cout<<endl;

system("pause");
return 0;

}
```

Zadanie 8 (C++) Napisz program rysujący kwadraty (rys. 2) rekurencyjnie jednym pociągnięciem kreski.



Rysunek 2.

Wskazówka: zmodyfikuj funkcję odpowiedzialną za rysowanie spirali z zadania **Z4**. *Istotny jest wybór właściwego miejsca wywołania rekurencyjnego.*

[p8.cpp]

```
#include <allegro.h>
#define m 800
#define n 600

void init();
void deinit();
```

```
void kwadraty(int x, int y, int xl, int yl);
char a[m+2][n+2], b[m+1][n+1];

BITMAP *bufor=NULL;

int main(){
      allegro_init();
      install_keyboard();
      set_color_depth(24);
      set_gfx_mode(GFX_AUTODETECT_WINDOWED,m,n,0,0);
      set_palette(default_palette);
      clear_to_color(screen,makecol24(0,244,255));
      bufor=create_bitmap(m,n);
    clear_to_color(bufor,makecol24(0,244,255));

    kwadraty(15,15,400,400);
    blit(bufor,screen,0,0,0,0,m,n);

    while(!key[KEY_ESC]){
        readkey();
        if(key[KEY_F9]){save_bitmap("kwadraty.bmp", bufor,
      default_palette);};
    }

    destroy_bitmap(bufor);
    deinit();
    return 0;
}
END_OF_MAIN()

void init() {
      int depth, res;
      allegro_init();
      depth = desktop_color_depth();
      if (depth == 0) depth = 32;
      set_color_depth(depth);
      res = set_gfx_mode(GFX_AUTODETECT_WINDOWED, 640, 480, 0, 0);
      if (res != 0) {
            allegro_message(allegro_error);
            exit(-1);
      }

      install_timer();
      install_keyboard();
      install_mouse();
      /* add other initializations here */
}

void deinit() {
      clear_keybuf();
      allegro_exit();
}

void kwadraty(int x, int y, int xl, int yl){
    int c=0xff0000;
    if(xl>2&&yl>2){
        hline(bufor,x,y,x+xl,c);//x1,y1,x2
        vline(bufor,x+xl,y,y+yl,c);//x1,y1,y2
        hline(bufor,x,y+yl,x+xl,c);
        vline(bufor,x,y+yl,y+yl/2,c);
```

```
        line(bufor,x,y+yl/2,x+xl/2,y+yl,c);
        line(bufor,x+xl/2,y+yl,x+xl,y+yl/2,c);
        line(bufor,x+xl,y+yl/2,x+xl/2,y,c);
        line(bufor,x+xl/2,y,x+xl/4,y+yl/4,c);

        x=x+xl/4;y=y+yl/4;xl=xl-xl/2;yl=yl-yl/2;
        kwadraty(x,y,xl,yl);

        xl=2*xl;yl=2*yl;x=x-xl/4;y=y-yl/4;
        line(bufor,x+xl/4,y+yl/4,x,y+yl/2,c);
        line(bufor,x,y+yl/2,x,y,c);
    }
}
```

Andrzej Pisarski