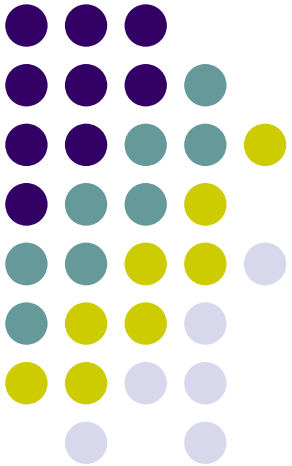
Średniozaawansowane programowanie w C++

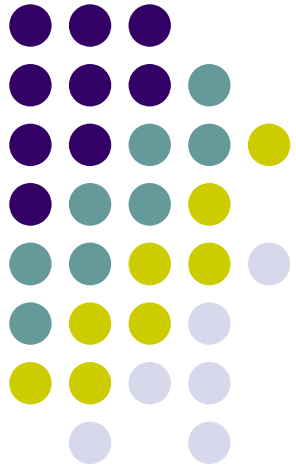
Wykład #5

18 kwietnia 2020 r.

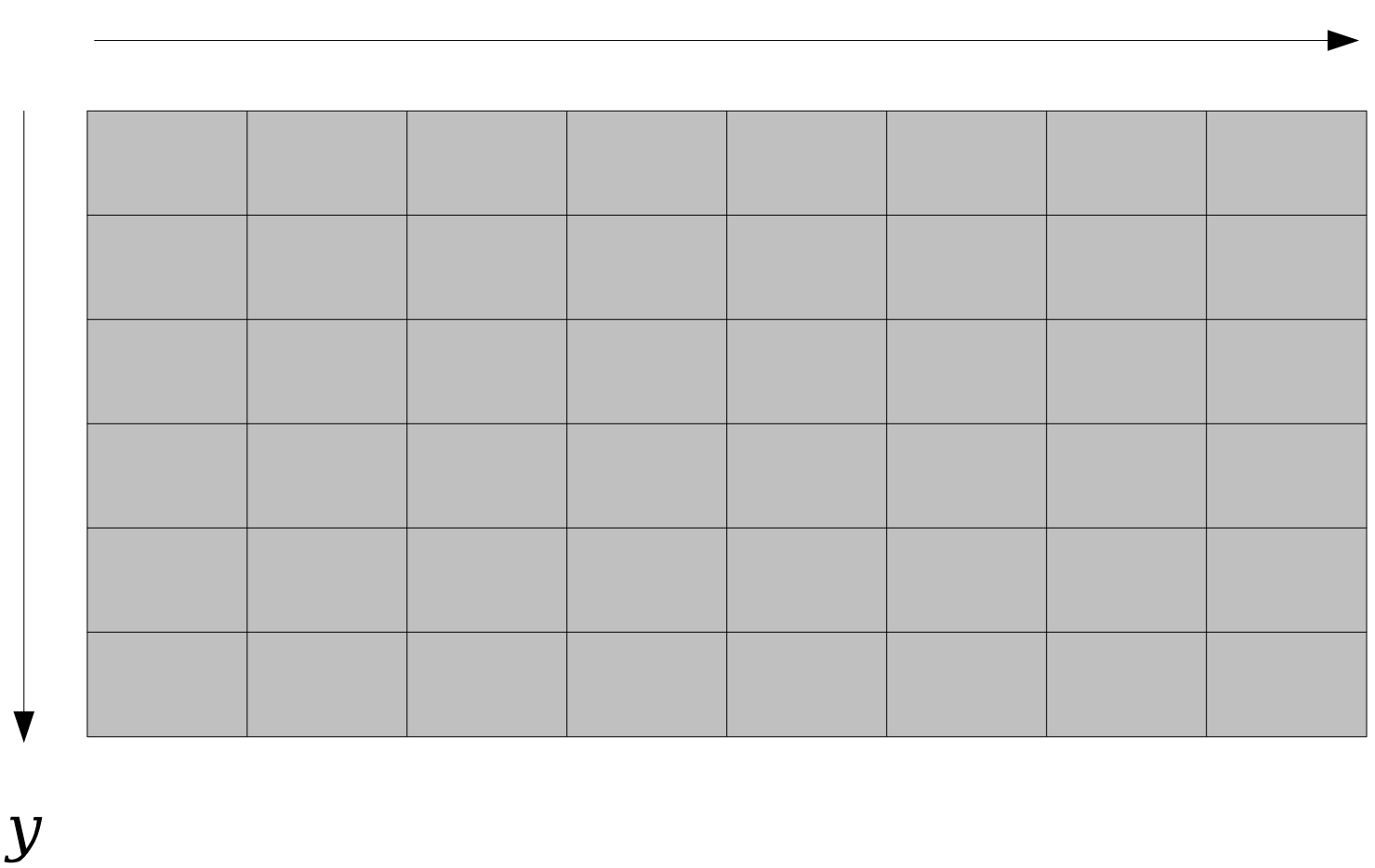


Grafika 2D

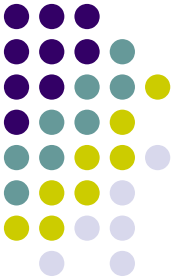
Chodź, pomaluj mój świat...



Ekran (8x6)



Kolory RGB

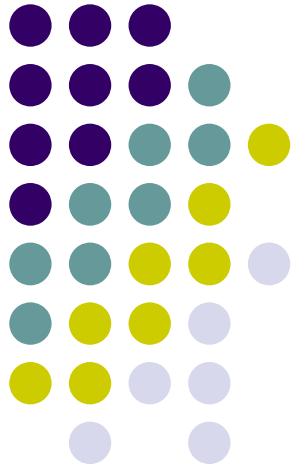


RGB = red + green + blue

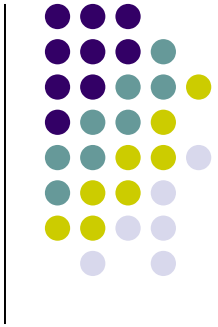
255, 0, 0	255, 0, 255	0, 0, 255
255, 255, 0	0, 255, 255	128, 128, 128
0, 255, 0	255, 255, 255	0, 0, 0

Allegro 4.4

Biblioteka graficzna

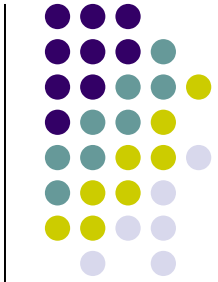


Użycie biblioteki



```
#include <allegro.h>

int main ()
{
        allegro_init ();         // inicjuje bibliotekę (OBLIGATORYJNE!)
        install_timer ();        // inicjuje zegar (m.in. przerwania)
        install_keyboard ();     // inicjuje klawiaturę
        install_mouse ();        // inicjuje mysz

        // Uruchamiamy grafikę w trybie pełnoekranowym 800x600, 16 bpp
        set_color_depth (16);    // ustawia bpp
        if (set_gfx_mode (GFX_AUTODETECT, 800, 600, 0, 0) != 0)
        {
                std::cerr << "Blad inicjowania trybu graficznego!";
                return 1;
        }
        set_window_title ("Tytul okna"); // ustawia tytuł okna

        // (…) coś sobie rysujemy

        allegro_exit ();                 // sprząta
        return 0;
}
END_OF_MAIN()   // makro robiące magię na koniec maina
```

Bitmapy



```
#include <allegro.h>

BITMAP *kwadrat_z_kropka ()
{
        const int ZIELONY = makecol (0, 255, 0);
        const int CZERWONY = makecol (255, 0, 0);
        BITMAP *bmp = create_bitmap (200, 200); // tworzymy bitmapę
        clear (bmp); // czyści bitmapę (na czarno)
        // albo: clear_to_color (kwadrat, makecol (255, 0, 0));
        rectfill (bmp, 50, 50, 100, 100, ZIELONY);      // prostokąt
        putpixel (bmp, 100, 100, CZERWONY);     // punkt
        return bmp;
}

// Gdzieś indziej w kodzie…
BITMAP *kwadrat = kwadrat_z_kropka ();
blit (kwadrat, screen, 0, 0, 0, 0, 200, 200);// umieszcza kwadrat na screen
destroy_bitmap (kwadrat);       // tak niszczymy bitmapy!
```

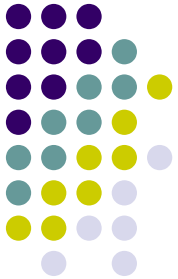
Więcej fantastycznych funkcji rysujących:
www.allegro.cc/manual/4/api/drawing-primitives/

Obsluga plikow graficznych

```
#include <allegro.h>

// Wczytujemy bitmapę (BMP) z pliku
BITMAP *obrazek = load_bitmap („kroliczek.bmp”, NULL);

// Edytujemy ją
dorysuj_spojrzenie_pelne_szalu (obrazek);

// Zapisujemy jako BMP
save_bmp („kroliczek.bmp”, obrazek, NULL);

// Pamiętamy o zwolnieniu bitmapy
destroy_bitmap (obrazek);

// Konwersja na PNG (używa pakietu imagemagic), 4 bpp (16 kolorów)
system („convert kroliczek.bmp -depth 4 kroliczek.png”);

// Usunięcie tymczasowego pliku BMP
remove („kroliczek.bmp”);
```

Co nieco o funkcjach odczytu/zapisu z/do plików:
www.allegro.cc/manual/4/api/loading-image-files/

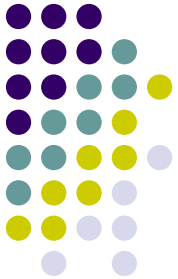
Tekst



```
#include <allegro.h>

void pisz_brzydkie_rzeczy (BITMAP *bmp)
{
    int lw = 8;
    textout_ex (bmp, font, "PASZTET!", 10, 10, makecol (0, 0, 255), -1);
    textprintf_centre_ex (bmp, font, 400, 30, makecol (0, 100, 243), -1,
        "Jestes brzydka jak %d wielorybow!", lw);
}
```

Opis funkcji wyświetlających tekst:
www.allegro.cc/manual/4/api/text-output/

Obsługa klawiatury

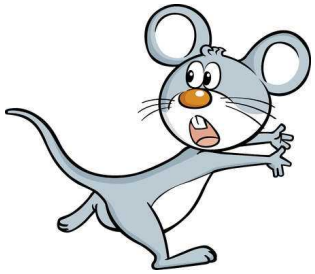


```
#include <allegro.h>

int klawisz = 0;
clear_keybuf ();        // opróżnia bufor klawiatury

while ((klawisz >> 8) != KEY_ESC)
{
        // (…) coś sobie rysujemy itd.
        klawisz = readkey ();
        if ((klawisz & 0xff) == 'k')
                /* wciśnięto małe k */ ;
        else if ((klawisz >> 8) == KEY_SPACE)
                /* wciśnięto spację */ ;
}

if (key [KEY_RCONTROL]) // czy w danej chwili jest wciśnięty prawy CTRL?
{
        // ZABIJ ZIEJĄCĄ OGNIEM POCZWARĘ!!!
}
```

Opis funkcji obsługi klawiatury:
www.allegro.cc/manual/4/api/keyboard-routines/
Lista oznaczeń klawiszy:
www.allegro.cc/manual/4/api/keyboard-routines/key

My się myszy nie boimy!

```
#include <allegro.h>

// Włączamy kursor myszy
show_mouse (screen);

// (…)

// Straszymy mysz przed rysowaniem
scare_mouse ();
```



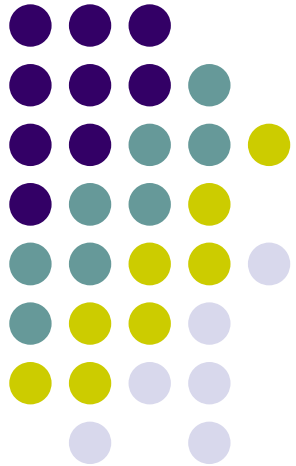
```
// Rysujemy na ekranie
circlefill (screen, 200, 200, 50, makecol (255, 255, 255));
```

```
// Wołamy mysz z powrotem
unscare_mouse ();
```

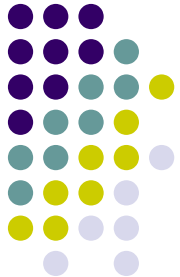
Opis funkcji obsługi myszy:
http://www.allegro.cc/manual/4/api/mouse-routines/

Przerwania sprzętowe

By żyło się lepiej…



Przerwanie klawiatury

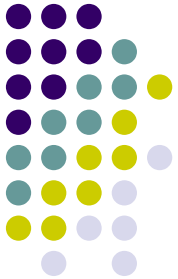


```
#include <allegro.h>

int przerwanie_klawiatury (int k)
{
        if ((k & 0xff) == 't')
        {
                // (…) zainicjuj opuszczanie pantografu
        }
        return 0; // klawisz został przechwycony
        // return k; // klawisz zostanie wysłany do bufora
}
END_OF_FUNCTION(przerwanie_klawiatury)

// (…) gdzieś w kodzie
LOCK_FUNCTION(przerwanie_klawiatury);
keyboard_callback = przerwanie_klawiatury;      // włącza przerwanie
```

UWAGA! Funkcja przerwania powinna wykonywać się bardzo szybko!

Więcej o przerwaniu klawiatury:
www.allegro.cc/manual/4/api/keyboard-routines/keyboard_callback

Przerwanie myszy



```
#include <allegro.h>

// Przerwanie myszy jest wołane przy każdej zmianie jej stanu
// (wciśnięcie/puszczenie przycisku, ruch wskaźnika)
void przerwanie_myszy (int m)
{
        if (m & MOUSE_FLAG_LEFT_UP)             // puszczony LPM
                zabij_potwora (mouse_x, mouse_y);
        else if (m & MOUSE_FLAG_RIGHT_UP)       // puszczony PPM
                zaznacz_obiekt (mouse_x, mouse_y);
}
END_OF_FUNCTION(przerwanie_myszy)

// (…) gdzieś w kodzie
show_mouse (screen);    // pokazuje kursor myszy

LOCK_FUNCTION(przerwanie_myszy);
mouse_callback = przerwanie_myszy;      // włącza przerwanie
```

UWAGA! Funkcja przerwania powinna wykonywać się naprawdę bardzo szybko!

Więcej o przerwaniu myszy:
www.allegro.cc/manual/4/api/mouse-routines/mouse_callback

Programowanie jest fantastyczne!!!