Министерство образования и науки Российской Федерации

Московский государственный технический университет им. Н. Э. Баумана

Кафедра «РК6»

**Отчет по лабораторной работе № 2**

По курсу «Программирование графических приложений»

**Вариант В12**

Выполнил: Петраков С.А.

Проверил: Родионов С. В.

Дата: 16.03.2021

Подпись: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2021 г.

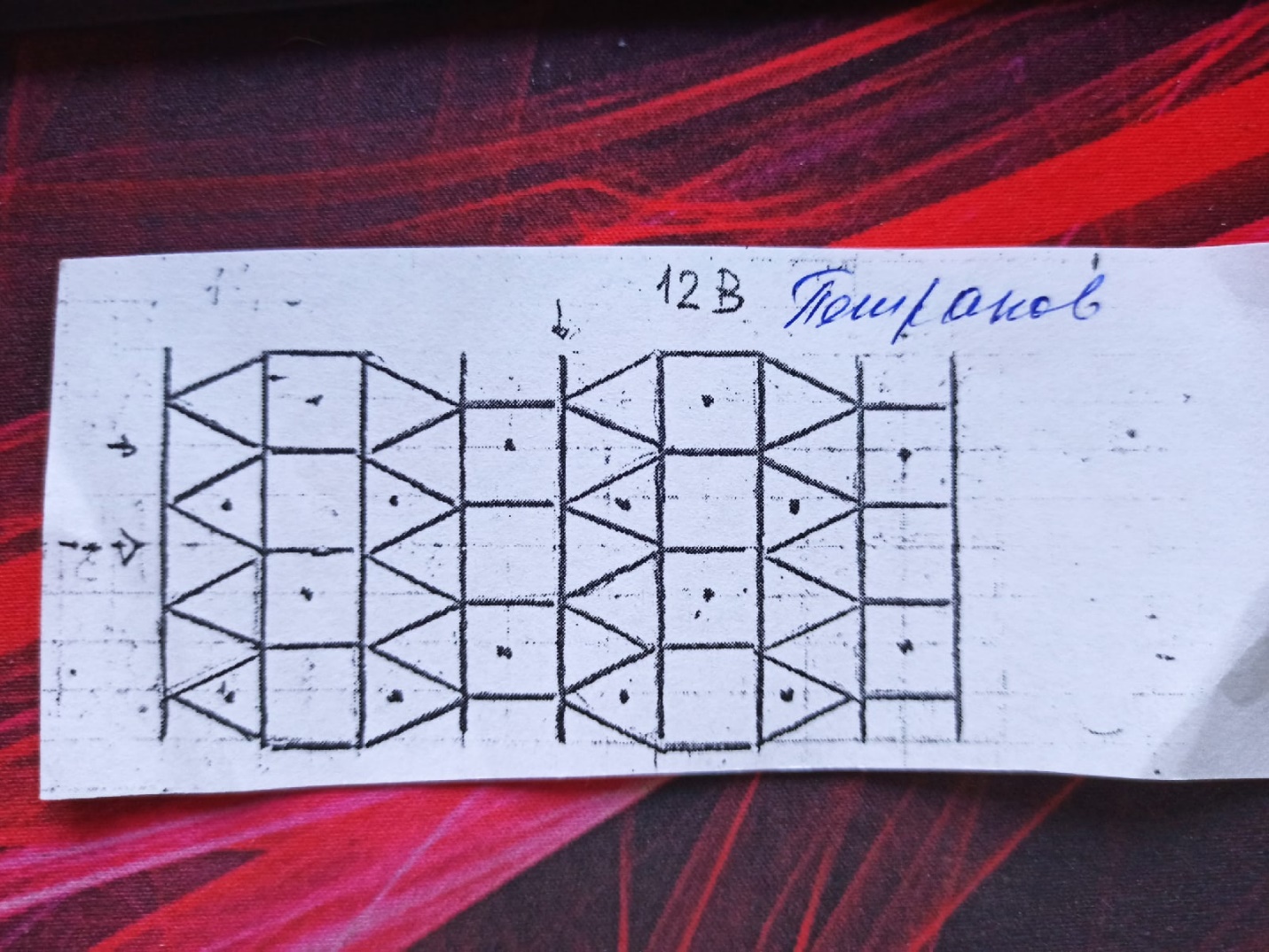
**Задание**

Разработать программу для создания графического окна, фон которого отображает заданный рисунок из периодически повторяющихся геометрических фигур. Их взаимное расположение должно образовать замощение плоскости графического окна независимо от его геометрических размеров. Программную реализацию такого замощения должна обеспечивать карта пикселей минимального по размерам прямоугольного фрагмента рисунка фона, который повторяется в нем. Такая пиксельная карта должна конструироваться в программе минимальным набором примитивов графического вывода и адресоваться графическому окну структурой его атрибутов. Это должно гарантировать автоматическое тиражирование карты пикселей слево-направо и сверху-вниз в пределах графического окна, пока оно отображено на экране. Кроме того, необходимо предусмотреть возможность интерактивно инвертировать цвет рисунка фона по нажатию и отпусканию любой кнопки мыши, когда ее курсор находится в границах графического окна. Для этого в программе нужно реализовать соответствующую обработку событий и изображений для ее графического окна. При разработке программы необходимо использовать библиотечные функции базисного программного интерфейса XWindowSystem.

**Выполнение работы**

Замощение плоскости (полу)правильными топологическими многоугольниками изучает геометрическая теория графов. На практике может быть построено бесконечно много их различных вариантов путем геометрической деформации многоугольных элементов. В частности, классическое замощение плоскости квадратами трансформируется в кольчужный узор из сопряженных колец, который показан на следующем рисунке:

Основная идея программной генерации фона графического окна Х-клиента состоит в том, чтобы подготовить пиксельную карту рисунка повторяющегося элемента фона и присвоить ее идентификатор фоновому атрибуту окна. Следующий листинг содержит исходный код программы генерации фона графического окна узора:



Согласно принципам структурного программирования функциональная часть этого исходного кода делится на 5 последовательных логических блоков: графическая инициализация, формирование пиксельной карты фона, создание графического окна и графический вывод.

В начальном блоке графической инициализации по запросу XopenDisplay адресуется структура дисплея. Затем устанавливается по умолчанию номер экрана, его корневое окно и число цветовых плоскостей, а также графический контекст изображения. Для получения этих дисплейных умолчаний вызываются соответствующие дисплейные макросы.

В следующем блоке формируется пиксельная карта повторяющегося элемента фона. При этом сначала по запросу XcreatePixmap создается пустая пиксельная карта требуемого размера 8х4 умноженные на размер клетки. Затем она заполняется белым квадратом по запросам Xforeground с белым макросом WhitePixelи XFillRectangle. После этого в ней рисуется требуемое изображение черной окружности по запросам Xforeground с черным макросом BlackPixel, XDrawLine и XDrawPoint. В общем случае для формирования изображения могут применяться различные примитивы графического вывода или растровые шаблоны из битовых карт.

В третьем блоке по запросу XcreateWindow создается графическое окно программы с заданным начальным размером 800х600 пикселей, которое является подокном корневого окна экрана, копирует его визуальный класс и глубину, а также декорируется оконным менеджером по атрибуту override\_redirect=False. Фоновый атрибут окна background\_pixmap фиксирует идентификатор пиксельной карты из предыдущего блока. Это гарантирует автоматическое заполнение фона окна рядами изображений пиксельной карты, независимо от изменений габаритов окна, когда оно отображается на экране. После создания окна, по запросу XselectInputу устанавливается маска получения событий нажатия (и отпускания) кнопок мыши и клавиш на клавиатуре, когда курсор мыши находится в габаритах окна или пересекает его рамку. В конце оконного блока сформированное окно отображается на экране по запросу XMapRaised, перекрывая другие окна. Кроме того, по запросу XSetFunctionв графическом контексте устанавливается функция инвертирования изображения для последующих операций графического вывода в цикле обработки событий.

Четвертый блок реализует цикл чтения и обработки событий из очереди Х-сервера, которые установлены для графического окна программы в блоке 3. Чтение поступающих событий обеспечивает запрос XNextEvent, а альтернативы их обработки выбираются по полю типа в структуре Xevent каждого полученного события. В частности, для событий нажатия и отпускания любой кнопки мыши, когда ее курсор находится внутри окна, предусмотрена следующая обработка. Сначала инвертируется изображение в пиксельной карте фона путем копирования ее области на себя по запросу XCopyAreaв графическом контексте с установленной функцией инвертирования. Затем инвертированная пиксельная карта переустанавливается для фона окна по запросу XSetWindowBackgroundPixmap. Наконец, очистка окна по запросу XclearWindow реализует инвертирование его фонового рисунка. Аналогичная обработка предусматривается для события пересечения курсором рамки окна в любом направлении. Итерации обработки этих мышиных событий могут продолжаться, пока флаг завершения цикла done=0 (False). В альтернативе обработки нажатия любой клавиши на клавиатуре (когда окно имеет фокус ввода) этому флагу присваивается значение ее физического кода из поля keycode структуры XEventполученного клавиатурного события. Его величина гарантированно больше 0 для любой клавиши. Поэтому значение флага doneстановится положительным (True) и происходит выход из цикла обработки событий для последующего завершения программы.

В блоке 5 графического выхода предусмотрен типовый набор завершающих действий. Сначала освобождается память пиксельной карты по запросу XFreePixmap. Затем по запросу XdestroyWindow закрывается окно программы. Наконец, запрос XcloseDisplay обеспечивает разрыв связи с X-сервером. Физический код клавиши, которая была нажата для выхода из цикла обработки событий возвращает exit-код программы.

Рассмотренный исходный текст программы генерации фона кольчуги можно приспособить для формирования фона с другим рисунком. Для этого нужно соответствующим образом изменить код рисования изображения в пиксельной карте повторяющегося фрагмента фона. После соответствующего редактирования текст программы следует сохранить с именем xfloor.cв текущем рабочем каталоге файловой системы. Его компиляцию и компановку с графической библиотекой обеспечивает следующая командная строка:

$ cc lab1.c –lX11

В результате будет получен выполняемый файл a.out, который может быть вызван для отображения на экране графического окна с заданным фоном.

**Код**

**“lab1.c”**

#include <X11/Xlib.h>

#define CELL 90

#define WIDTH CELL\*8 /\* repeat element width \*/

#define HEIGHT CELL\*4 /\* repeat element height \*/

int main(int argc, char\* argv[]) {

Display\* dpy; /\* display structure \*/

int src; /\* screen number \*/

GC gc; /\* graphic context \*/

int depth; /\* bit per color pixel \*/

Window root; /\* screen root window \*/

Pixmap pix; /\* Pixel map for ring \*/

Window win; /\* Main window \*/

XSetWindowAttributes attr; /\* window attributes \*/

unsigned long mask; /\* event mask \*/

XEvent event; /\* Event structure \*/

unsigned int done = 0; /\* exit code \*/

/\* X init \*/

dpy = XOpenDisplay(NULL);

src = DefaultScreen(dpy);

depth = DefaultDepth(dpy, src);

root = DefaultRootWindow(dpy);

gc = DefaultGC(dpy, src);

/\* Pixmap block \*/

pix = XCreatePixmap(dpy, root, WIDTH, HEIGHT, depth);

XSetBackground(dpy, gc, WhitePixel(dpy, src));

XSetForeground(dpy, gc, WhitePixel(dpy, src));

XFillRectangle(dpy, pix, gc, 0, 0, WIDTH, HEIGHT);

XSetForeground(dpy, gc, BlackPixel(dpy, src));

XDrawLine(dpy, pix, gc, CELL \* 0, CELL \* 0, CELL \* 0, CELL \* 4);

XDrawLine(dpy, pix, gc, CELL \* 0, CELL \* 1, CELL \* 2, CELL \* 0);

XDrawLine(dpy, pix, gc, CELL \* 0, CELL \* 1, CELL \* 2, CELL \* 2);

XDrawLine(dpy, pix, gc, CELL \* 0, CELL \* 3, CELL \* 2, CELL \* 2);

XDrawLine(dpy, pix, gc, CELL \* 0, CELL \* 3, CELL \* 2, CELL \* 4);

XDrawLine(dpy, pix, gc, CELL \* 2, CELL \* 0, CELL \* 2, CELL \* 4);

XDrawLine(dpy, pix, gc, CELL \* 2, CELL \* 0, CELL \* 4, CELL \* 0);

XDrawLine(dpy, pix, gc, CELL \* 2, CELL \* 2, CELL \* 4, CELL \* 2);

XDrawLine(dpy, pix, gc, CELL \* 4, CELL \* 0, CELL \* 4, CELL \* 4);

XDrawLine(dpy, pix, gc, CELL \* 4, CELL \* 0, CELL \* 6, CELL \* 1);

XDrawLine(dpy, pix, gc, CELL \* 6, CELL \* 1, CELL \* 4, CELL \* 2);

XDrawLine(dpy, pix, gc, CELL \* 4, CELL \* 2, CELL \* 6, CELL \* 3);

XDrawLine(dpy, pix, gc, CELL \* 6, CELL \* 3, CELL \* 4, CELL \* 4);

XDrawLine(dpy, pix, gc, CELL \* 6, CELL \* 0, CELL \* 6, CELL \* 4);

XDrawLine(dpy, pix, gc, CELL \* 6, CELL \* 1, CELL \* 8, CELL \* 1);

XDrawLine(dpy, pix, gc, CELL \* 6, CELL \* 3, CELL \* 8, CELL \* 3);

XDrawPoint(dpy, pix, gc, CELL \* 3, CELL \* 1);

XDrawPoint(dpy, pix, gc, CELL \* 1, CELL \* 3);

XDrawPoint(dpy, pix, gc, CELL \* 5, CELL \* 3);

XDrawPoint(dpy, pix, gc, CELL \* 7, CELL \* 2);

/\* Window block \*/

mask = (CWOverrideRedirect | CWBackPixmap);

attr.override\_redirect = False;

attr.background\_pixmap = pix;

win = XCreateWindow(dpy, root, 0, 0, 800, 600, 1, depth,

InputOutput, CopyFromParent, mask, &attr);

mask = (ButtonPressMask | ButtonReleaseMask |

EnterWindowMask | LeaveWindowMask |

KeyPressMask);

XSelectInput(dpy, win, mask);

XMapRaised(dpy, win);

XSetFunction(dpy, gc, GXinvert);

/\* Dispatch block \*/

while (done == 0) {

XNextEvent(dpy, &event);

switch (event.type) {

case EnterNotify:

case LeaveNotify:

case ButtonPress:

case ButtonRelease:

XCopyArea(dpy, pix, pix, gc, 0, 0, WIDTH, HEIGHT, 0, 0);

XSetWindowBackgroundPixmap(dpy, win, pix);

XClearWindow(dpy, win);

break;

case KeyPress:

done = event.xkey.keycode;

break;

default: break;

}

}

/\* X-Exit block \*/

XFreePixmap(dpy, pix);

XDestroyWindow(dpy, win);

XCloseDisplay(dpy);

return(done);

}

**Литература**

1. Конспекты лекций

2. http://bigor.bmstu.ru

3. Теория графов, Омельченко А.В., 2018

4. 09-02-2021-Электронныи\_\_учебник\_Программирование\_в\_X\_Window\_

System.htm\_\_