BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH**

**BÀI TẬP LỚN MÔN LẬP TRÌNH ĐỒ HỌA**

**GAME CANDY RUSH**

**BÁO CÁO BÀI TẬP LỚN**

**Ngành: Khoa Học Máy Tính / Công Nghệ Thông Tin**

**Lớp: CS81**

**Sinh viên thực hiện:**

**Nguyễn Thị Diễm My – 1851050091**

**Nguyễn Duy Minh Hoàng – 1651012058**

**Nguyễn Hồng Phương Uyên - 1651010161**

**Giảng viên giảng dạy: ThS. VÕ THỊ HỒNG TUYẾT**

**TP. HỒ CHÍ MINH, 02/2022**

LỜI MỞ ĐẦU

Ngày nay, ngành công nghiệp Game đang trên đà phát triển rất mạnh mẽ. Game không chỉ là phương tiện giải trí cho người sử dụng mà còn là nguồn sáng tạo không ngừng từ nhà sản xuất. Từ đó thôi thúc rất nhiều Game hay và hấp dẫn ra đời. Thế nhưng để có thể cho ra đời một sản phẩm Game thì sẽ phải thực hiện bằng một ngôn ngữ lập trình.

Sẽ không có gì đáng ngạc nhiên khi thấy C++ là ngôn ngữ lập trình Game phổ biến bởi khả năng kiểm soát các thành phần hệ thống cấp thấp đã góp phần kéo dài thời gian chạy chương trình – một yếu tố rất cần thiết trong lập trình Game.

Từ xu hướng phát triển và sáng tạo không ngừng, bài tập lớn này sẽ nghiên cứu để xây dựng trò chơi Candy Rush với các hình ảnh đồ họa tự thiết kế từ Photoshop, ngôn ngữ C++ và thư viện OpenGL.

**MỤC LỤC**

[Chương 1. GIỚI THIỆU 1](#_Toc95166967)

[1.1. Giới thiệu đề tài 1](#_Toc95166968)

[1.2. Mục tiêu đề tài 1](#_Toc95166969)

[1.3. Giới hạn đề tài 1](#_Toc95166970)

[1.4. Bố cục báo cáo 1](#_Toc95166971)

[Chương 2. GAME CANDY RUSH 3](#_Toc95166972)

[2.1. Các kiến thức cơ bản 3](#_Toc95166973)

[2.1.1. OpenGL 3](#_Toc95166974)

[2.1.2. Một số thành phần cơ bản lập trình đồ họa 4](#_Toc95166975)

[2.1.3. Các hàm shape 5](#_Toc95166976)

[2.1.4. Sự kiện chuột 7](#_Toc95166977)

[2.1.5. Sự kiện bàn phím 8](#_Toc95166978)

[2.2. Thuật giải đề xuất 8](#_Toc95166979)

[2.2.1. Thiết kế hình ảnh đồ họa 8](#_Toc95166980)

[2.2.2. Sắp xếp kẹo 12](#_Toc95166981)

[2.2.3. Hiển thị menu game 12](#_Toc95166982)

[2.2.4. Hoán đổi kẹo 13](#_Toc95166983)

[2.2.5. Di chuyển máy ảnh 14](#_Toc95166984)

[2.2.6. Lượt hoán đổi kẹo 14](#_Toc95166985)

[2.2.7. Tính điểm 15](#_Toc95166986)

[2.3. Kết quả 15](#_Toc95166987)

[Chương 3. KẾT LUẬN 16](#_Toc95166988)

[3.1. Kết quả đạt được 16](#_Toc95166989)

[3.1.1. Các công cụ hỗ trợ 16](#_Toc95166990)

[3.1.2. Kết quả 16](#_Toc95166991)

[3.2. Kết luận 20](#_Toc95166992)

[3.2.1. Ưu điểm 20](#_Toc95166993)

[3.2.2. Nhược điểm 20](#_Toc95166994)

[3.3. Định hướng phát triển trong tương lai 21](#_Toc95166995)

DANH MỤC TỪ VIẾT TẮT

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Từ viết tắt | Từ đầy đủ | Diễn giải |
| 1 | API | Application Programming Interface | phương thức trung gian kết nối các thư viện và ứng dụng khác nhau |
| 2 | OpenGL | Open Graphics Library | một API đa nền tảng, đa ngôn ngữ cho render đồ họa vector 2D và 3D |

DANH MỤC HÌNH ẢNH

[Hình 2‑1 OpenGL 4](#_Toc95166936)

[Hình 2‑2 Wire Shapes 6](#_Toc95166937)

[Hình 2‑3 Solid Shapes 7](#_Toc95166938)

[Hình 2‑4 Viên kẹo 1 9](#_Toc95166939)

[Hình 2‑5 Viên kẹo 2 9](#_Toc95166940)

[Hình 2‑6 Viên kẹo 3 10](#_Toc95166941)

[Hình 2‑7 Viên kẹo 4 10](#_Toc95166942)

[Hình 2‑8 Viên kẹo 5 11](#_Toc95166943)

[Hình 2‑9 Viên kẹo 6 11](#_Toc95166944)

[Hình 2‑10 Viên kẹo 7 12](#_Toc95166945)

[Hình 2‑11 Màn hình nền 12](#_Toc95166946)

[Hình 2‑12 Sơ đồ Menu Game 13](#_Toc95166947)

[Hình 2‑13 Lưu đồ nhấp chuột trái hoán đổi kẹo 13](#_Toc95166948)

[Hình 3‑1 Màn hình đầu tiên 16](#_Toc95166949)

[Hình 3‑2 Menu vào game 17](#_Toc95166950)

[Hình 3‑3 Giao diện chơi game 18](#_Toc95166951)

[Hình 3‑4 Menu trong game 19](#_Toc95166952)

[Hình 3‑5 Game over 20](#_Toc95166953)

DANH MỤC BẢNG

# GIỚI THIỆU

## Giới thiệu đề tài

Với sự phát triển của ngành công nghiệp giải trí như trò chơi, ngày càng có nhiều nhu cầu về lập trình đồ họa. Candy Rush là một trò chơi điện tử thu hút được lượng người chơi khổng lồ. Nhiệm vụ của người chơi là phải ghép từ 3 viên kẹo cùng loại để làm chúng biến mất và vượt qua.

Đề tài “Game Candy Rush” được thực hiện thông qua Visual Studio 2017/2019 bằng ngôn ngữ C++ và sự hỗ trợ của thư viện OpenGL là cốt lõi cùng với thư viện bên ngoài giúp hiển thị hình ảnh. Các hình ảnh trong game như nền, kẹo,… được thiết kế từ ứng dụng Photoshop.

## Mục tiêu đề tài

Đề tài có những mục tiêu cụ thể như sau:

* Lập trình đồ họa một game cơ bản
* Các hình ảnh đồ họa tự thiết kế
* Sử dụng ngôn ngữ C++
* Sử dụng thư viện OpenGL là cốt lõi
* Hiển thị hình ảnh lên màn hình đồ họa
* Thao tác được trò chơi trên màn hình đồ họa

## Giới hạn đề tài

Về nội dung: tự thiết kế hình ảnh đồ họa và xây dựng trò chơi Candy Crush cơ bản.

Về công nghệ: sử dụng ngôn ngữ lập trình C++ với thư viện OpenGL với Visual Studio 2017/2019.

## Bố cục báo cáo

Nội dung báo cáo gồm 3 chương, cụ thể như sau:

* Chương 1. Giới thiệu: giới thiệu, nêu lên mục tiêu, giới hạn và bố cục đề tài.
* Chương 2. Game Candy Rush: trình bày các kiến thức cơ bản, thuật giải đề xuất và kết quả.
* Chương 3. Kết luận: kết quả đạt được, rút ra ưu nhược điểm và cho biết định hướng phát triển trong tương lai.

# GAME CANDY RUSH

## Các kiến thức cơ bản

### OpenGL

OpenGL được phát triển bởi công ty Silicon Graphics, Inc. Đây là một API đa nền tảng, đa ngôn ngữ cho render đồ họa vector 2D và 3D. API thường được sử dụng để tương tác với bộ xử lý đồ họa, nhằm đạt được tốc độ render phần cứng. Cụ thể như sau:

* “Cổng giao tiếp (interface) phần mềm với phần cứng đồ họa”.
* Thư viện đồ họa 3D và mô hình hóa có tính chuyển đổi (portable) cao và tốc độ thể hiện nhanh.
* Tạo các đối tượng 3D với tốc độ nhanh nhưng chất lượng hình ảnh gần như chất lượng của kỹ thuật ray-tracer.
* Không phải là một ngôn ngữ lập trình. Ngôn ngữ lập trình gốc làm việc là C. Hiện nay, có thể sử dụng trong các ngôn ngữ khác nhưng phải nhờ vào một số kỹ thuật trung gian.
* Gồm các thư viện động nằm trong thư mục hệ thống của Windows và các file header.
* Thiết kế độc lập hệ điều hành, code sử dụng có thể chuyển đổi qua lại giữa các hệ điều hành.
* Độc lập platform nên không cung cấp các hàm quản lý cửa sổ và màn hình cũng như các hàm xử lý sự kiện chuột và bàn phím.
* Khi thao tác với giao diện, cần sử dụng các thư viện tiện ích hoặc chính thư viện đồ họa của hệ điều hành.



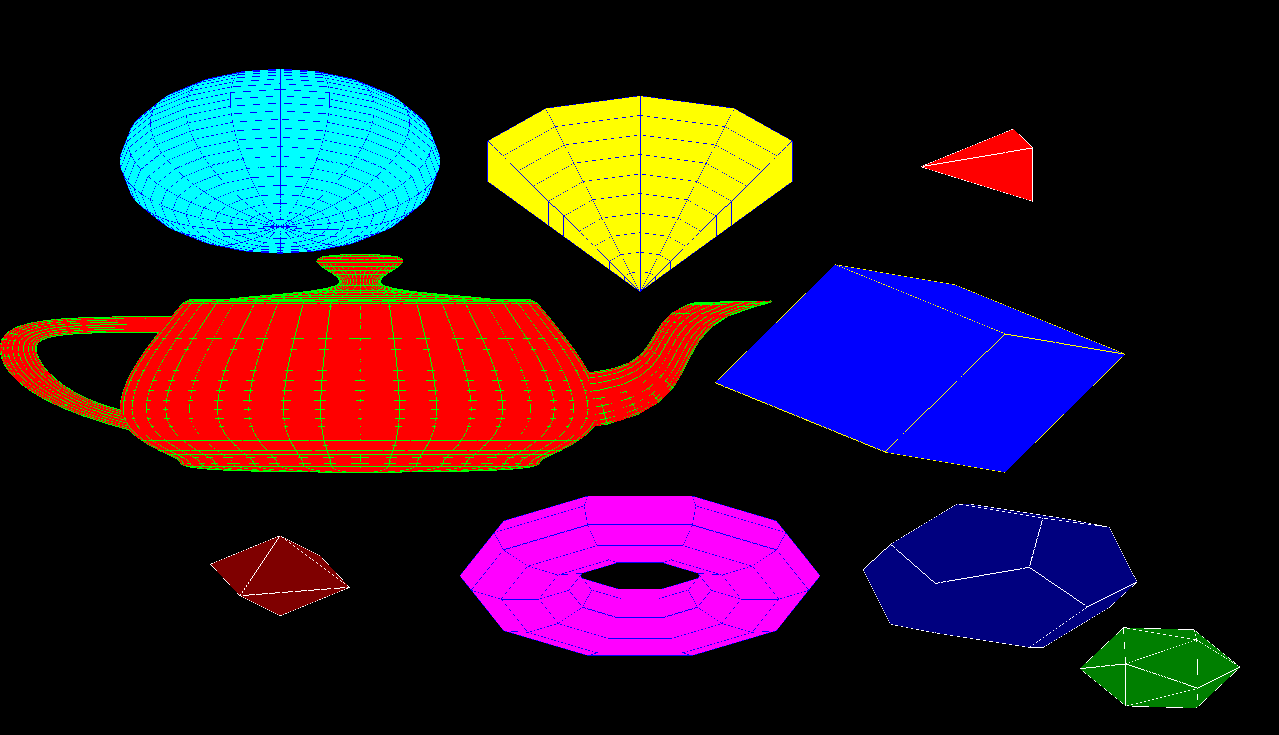
Hình 2‑1 OpenGL

### Một số thành phần cơ bản lập trình đồ họa

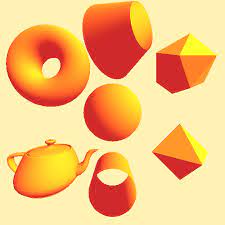
* Khởi tạo window:
* Tạo cửa sổ window int glutCreateWindow(char \*name); trong đó: name là chuỗi xác định tên của window
* Khởi tạo OpenGL
* Chọn màu nền glClearColor(GLfloat red, GLfloat green, GLfloat blue, GLfloat alpha);
* Đăng ký các hàm callback:
* hàm hiển thị: hàm đăng ký void glutDisplayFunc(void (\*func)(void)); hàm hiển thị void RenderScene(void);
* hàm xử lý sự kiện window thay đổi kích thước: hàm đăng ký void glutReshapeFunc(void (\*func)(int width, int height)); hàm xử lý sự kiện window thay đổi kích thước void ChangeSize(GLsizei w, GLsizei h);
* hàm xử lý các sự kiện bàn phím: hàm đăng ký void glutKeyboardFunc(void (\*func)(unsigned char key, int x, int y)); hàm xử lý các sự kiện bàn phím void keyboard(unsigned char key, int x, int y);
* hàm xử lý các sự kiện chuột: hàm đăng ký void glutMouseFunc(void (\*func)(int button, int state, int x, int y));
* hàm định thời: hàm đăng ký void glutTimerFunc(unsigned int msecs, void (\*func)(int value), value); hàm callback void TimerFunction(int value);
* Xóa bộ đệm màu và depth buffer:
* glClear(GL\_COLOR\_BUFFER\_BIT|GL\_DEPTH\_BUFFER\_BIT );
* Màu của bộ đệm màu được gán bằng giá trị của màu nền
* Giá trị của depth buffer được gán giá trị nhỏ nhất
* Xuất ra màn hình:
* Nếu chọn kiểu hiển thị là GLUT\_SINGLE sử dụng glFlush();
* Nếu chọn kiểu hiển thị là GLUT\_DOUBLE sử dụng glutSwapBuffers();
* Xác định viewport:
* void glViewport( GLint x, GLint y, GLsizei width, GLsizei height ); trong đó (x,y) là tọa độ góc trái dưới của viewport, (width, height) là kích thước của viewport
* Xác định phép chiếu:
* void glOrtho( GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble zNear, GLdouble zFar );

### Các hàm shape

* Cube:
* glutSolidCube(size)
* glutWireCube(size)
* Cone:
* glutSolidCone(base\_radius, height, slides, stacks)
* glutWireCone(base\_radius, height, slides, stacks)
* Sphere:
* glutSolidSphere(radius, slides, stacks)
* glutWireSphere(radius, slides, stacks)
* Torus:
* glutSolidTorus(linner\_radius, outer\_radius, nsides, rings)
* glutWireTorus(linner\_radius, outer\_radius, nsides, rings)



Hình 2‑2 Wire Shapes



Hình 2‑3 Solid Shapes

### Sự kiện chuột

Mouse sử dụng rất phổ biến và không thể thiếu. Với Game: Mouse sử dụng để định dạng phương hướng nhìn của model chủ vào các model khác, hoặc nhận nhiệm vụ là bắn đạn, tấn công của các nhân vật game….

Bắt sự kiện chuột trái, chuột phải và chuột giữa:

Hàm: void MouseButton(int type\_button, int state, int x, int y);

Trong đó:

* type\_button là kiểu phím bấm chuột, đó thể hiện là chuột trái hay chuột phải. Có 3 loại cho type\_button được định nghĩa trong glut.h như sau

#define GLUT\_LEFT\_BUTTON 0

#define GLUT\_MIDDLE\_BUTTON 1

#define GLUT\_RIGHT\_BUTTON 2

* state là trạng thái bấm xuông hay nhả ra
* x,y là tọa độ hiện tại của chuột trên màn hình windows.

Gọi hàm MouseButton trong hàm main thông qua hàm xử lý chuột: glutMouseFunc();

### Sự kiện bàn phím

* Bắt sự kiện của các phím chức năng như Left Right Up Down, F1-F12…

+ 12 phím F đi từ F1-F12

+ 4 phím : Up Down Left Right

+ Phím: Insert/End/Page Down-Up/ Home

+ glutSpecialFunc(//);

* Bắt sự kiện của các ký tự A B C…

+ glutKeyboardFunc(//);

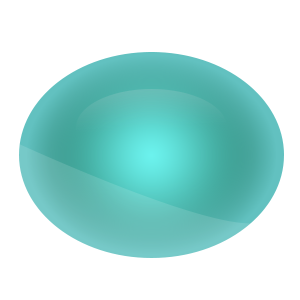
+ glutSpecialUpFunc(releaseKey);

+ glutKeyboardUpFunc(//);

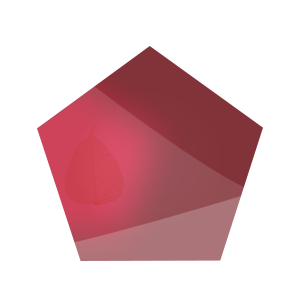
## Thuật giải đề xuất

### Thiết kế hình ảnh đồ họa

Sử dụng ứng dụng photoshop để tạo ra 7 viên kẹo và màn hình nền rồi xây dựng mô hình 3d theo thiết kế



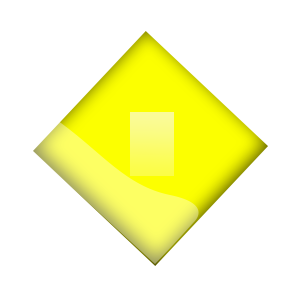
Hình 2‑4 Viên kẹo 1



Hình 2‑5 Viên kẹo 2



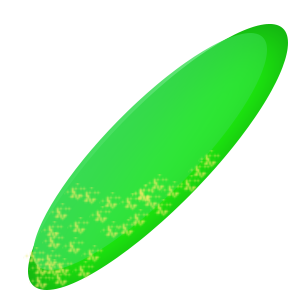
Hình 2‑6 Viên kẹo 3



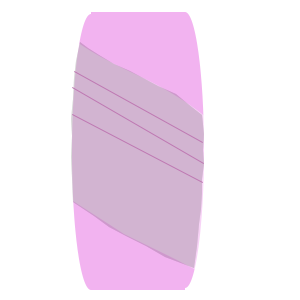
Hình 2‑7 Viên kẹo 4



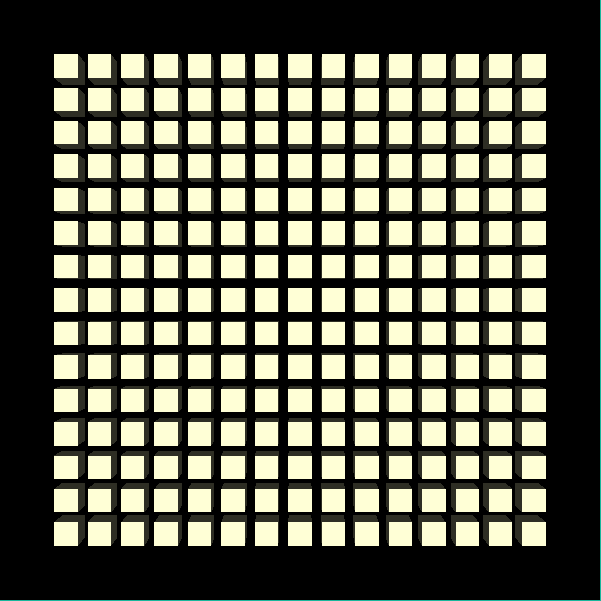
Hình 2‑8 Viên kẹo 5



Hình 2‑9 Viên kẹo 6



Hình 2‑10 Viên kẹo 7



Hình 2‑11 Màn hình nền

### Sắp xếp kẹo

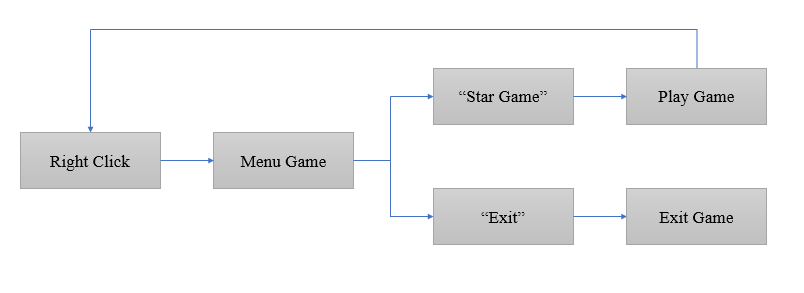
Các viên kẹo được sắp xếp ngẫu nhiên vào lưới 15x15:

* sử dụng vòng lặp for lồng for cho ma trận 15x15
* mỗi ô sử dụng hàm rand() lấy giá trị từ 1 đến 7 (tương ứng 7 loại kẹo)

### Hiển thị menu game

Sử dụng sự kiện chuột phải:

* hàm glutCreateMenu
* Start Game: bắt đầu chơi hoặc chơi lại
* Exit: thoát game

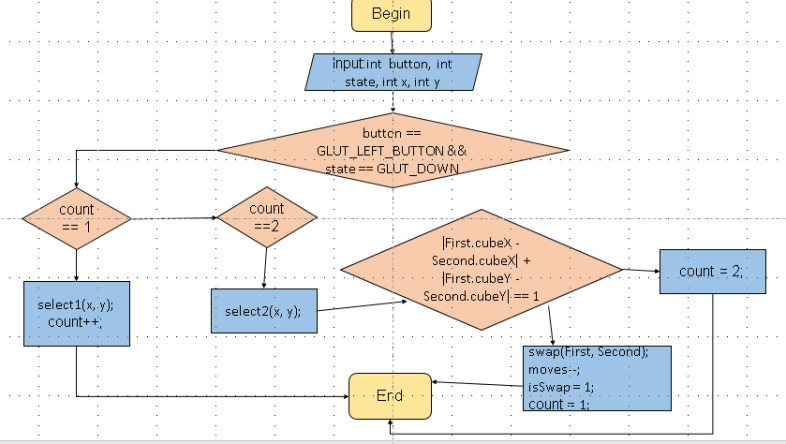


Hình 2‑12 Sơ đồ Menu Game

### Hoán đổi kẹo

Sử dụng sự kiện chuột trái:

* đổi vị trí giữa viên kẹo nhấp trước và nhấp sau: cho globalMouseCounter = 1 hoặc 2 để phân biệt
* chỉ có thể đổi vị trí trên, dưới, trái và phải nếu nhấp khác thì nhấp chuột sẽ hủy



Hình 2‑13 Lưu đồ nhấp chuột trái hoán đổi kẹo

### Di chuyển máy ảnh

Sử dụng sự kiện bàn phím để thực hiện: các phím chức năng như Left Right Up Down, F1 F2 F4

* F1: thu nhỏ lưới
* xpos giảm theo sin(y)
* ypos tăng theo sin(x)
* zpos tăng theo cos(y)
* F2: phóng to lưới:
* xpos tăng theo sin(y)
* ypos giảm theo sin(x)
* zpos giảm theo cos(y)
* F4: đặt máy ảnh về chế độ xem ban đầu:
* đặt các giá trị rot, pos, góc về 0
* Up: di chuyển máy ảnh lên trên
* xrot giảm đi 1
* nếu xrot bé hơn -360 thì tăng thêm 360
* Down: di chuyển máy ảnh lên trên
* xrot tăng thêm 1
* nếu xrot lớn hơn 360 thì giảm đi 360
* Left: di chuyển máy ảnh chếch sang trái
* yrot giảm đi 1
* nếu yrot bé hơn -360 thì tăng thêm 360
* Right: di chuyển máy ảnh chếch sang phải
* yrot tăng thêm 1
* nếu yrot lớn hơn 360 thì giảm đi 360

### Lượt hoán đổi kẹo

* Giới hạn số lượt hoán đổi kẹo là 30
* Mỗi lần “Star Game” thì đặt moves = 30
* Mỗi lần hoán đổi thành công thì moves giảm 1
* Khi moves = 0 thì dừng game và hiển thị “Game Over”

### Tính điểm

* Mỗi lần “Star Game” thì đặt score = 0
* Sau mỗi lần hoán đổi nếu xếp được thành hàng có từ 3 viên kẹo trở lên cùng loại thì tăng score với mỗi viên +10
* Hàng kẹo khi được tính điểm sẽ cập nhật kẹo mới vào
* Khi lượt đi hết “Game Over” thì score hiển thị sẽ là số điểm cuối cùng

## Kết quả

Dựng chương trình Game Candy Crush với:

* 1 màn có 30 lượt đi
* +10 cho mỗi viên kẹo thì xếp hàng thành công
* Menu game để bắt đầu hoặc thoát game
* Thay đổi các hướng nhìn trong không gian 3D

# KẾT LUẬN

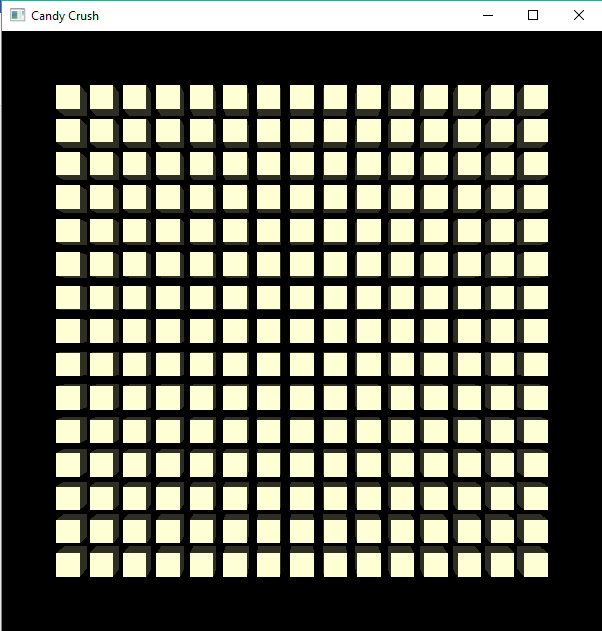
## Kết quả đạt được

### Các công cụ hỗ trợ

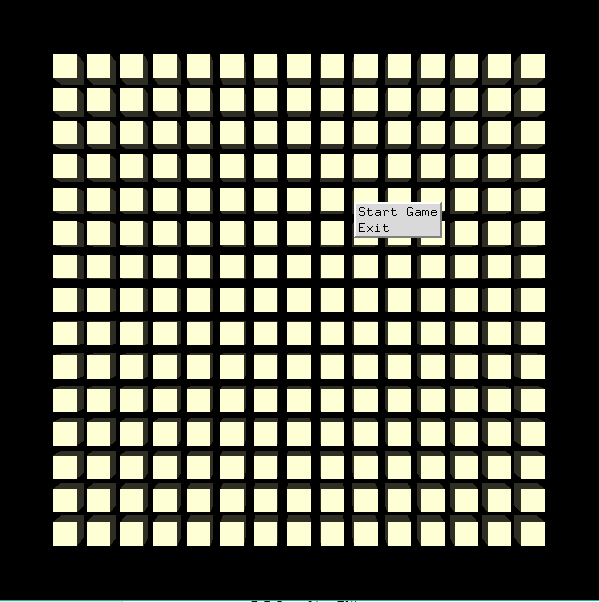
Sử dụng phần mềm Microsolf Visual Studio 2017/2019 lập trình C++. Dùng OpenGL để hỗ trợ hiển thị hình ảnh kết quả.

### Kết quả

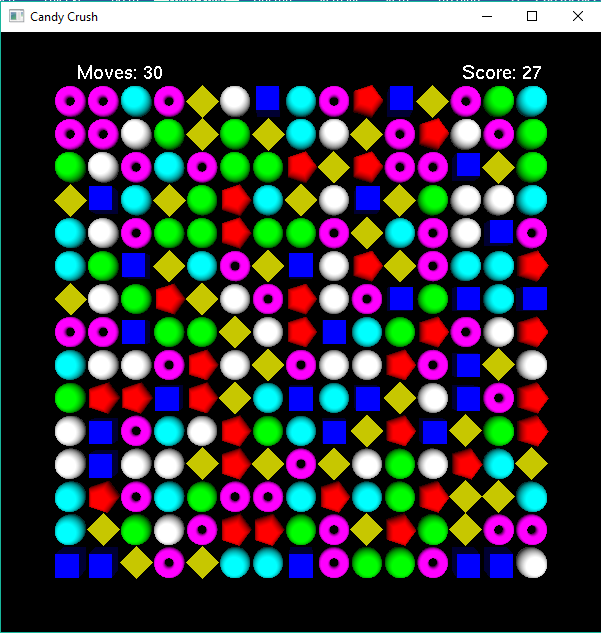
Đề tài đạt được những kết quả khi chạy demo như sau:



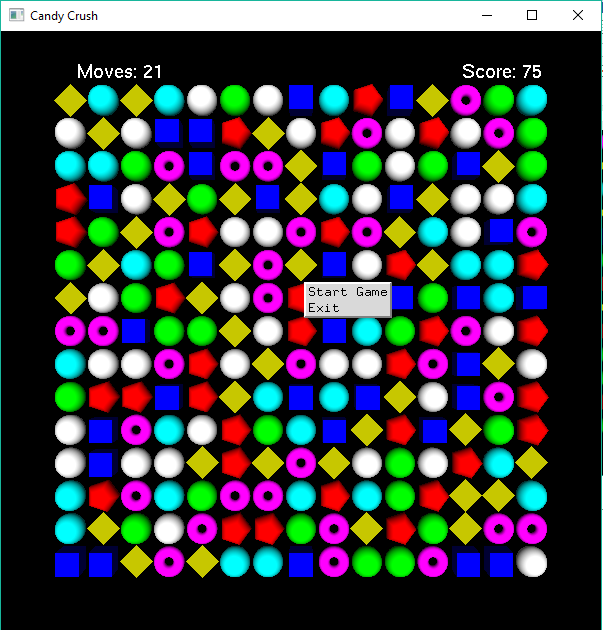
Hình 3‑1 Màn hình đầu tiên



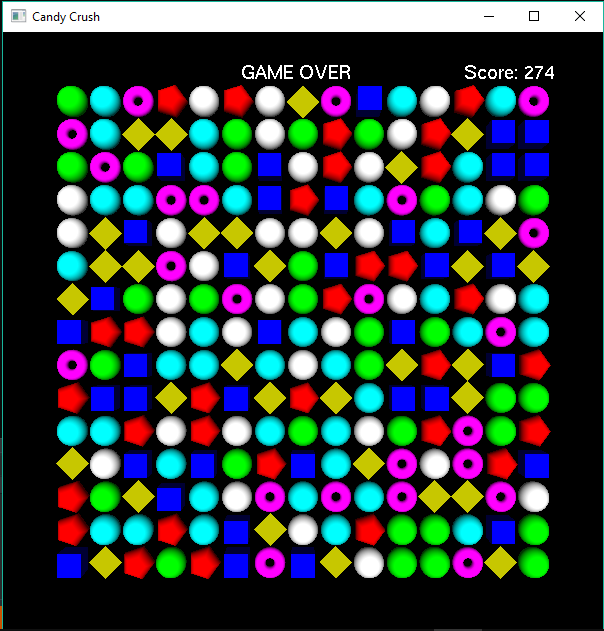
Hình 3‑2 Menu vào game



Hình 3‑3 Giao diện chơi game



Hình 3‑4 Menu trong game



Hình 3‑5 Game over

## Kết luận

### Ưu điểm

Đề tài có những ưu điểm sau:

* Ôn lại các kiến thức về lập trình đồ họa 2D và 3D
* Đề tài về Game gây thú vị
* Tìm hiểu một số giải thuật

### Nhược điểm

Đề tài có những nhược điểm sau:

* Game có nhiều tính năng và cấp độ chưa thực hiện được
* Việc tính toán hiển thị chiếm dụng khá nhiều tài nguyên
* Thuật toán khá phức tạp
* Thiếu animation
* Đồ họa đơn giản

## Định hướng phát triển trong tương lai

Một số định hướng phát triển trong tương lai:

* Hoàn thiện tất cả các tính năng và cấp độ
* Tạo ra các mô hình 3D đẹp mắt, xử lý để trở nên mềm và chân thực như ngoài thế giới tự nhiên.
* Triển khai các hiệu ứng, theo dõi chi tiết hơn sự thay đổi sắc độ của các viên kẹo với chuyển động nguồn sáng.
* Tối ưu hóa chi phí tính toán và thời gian chạy.

Tài liệu tham khảo

|  |  |
| --- | --- |
| [1] | V. T. H. Tuyết, LẬP TRÌNH ĐỒ HỌA, Hồ Chí Minh, 2020. |
| [2] | Silicon Graphics Incorporated, "OpenGL examples," [Online]. Available: https://www.opengl.org/archives/resources/code/samples/glut\_examples/examples/examples.html. [Accessed 2021]. |