TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP.HCM
KHOA CÔNG NGHỆ THÔNG TIN
MÔN: **PHÁT TRIỂN GAME**
LỚP: 16CNTN

ĐỒ ÁN GIỮA KỲ

GAME HALLOWEEN DEFENSE

NHÓM: RANGER

1612829 – Nguyễn Quốc Vương

1612842 – Lê Thành Công

GIẢNG VIÊN LÝ THUYẾT

Trần Minh Triết

GIẢNG VIÊN THỰC HÀNH

Trần Ngọc Đạt Thành

Mục lục

I. Thông tin nhóm

- Tên nhóm: **Ranger**

STT	MSSV	Họ và tên	Email
1	1612829	Nguyễn Quốc Vương	1612829@student.hcmus.edu.vn
2	1612842	Lê Thành Công	1612842@student.hcmus.edu.vn

- Tên game: **Halloween Defense**

II. Cốt truyện

Vào một đêm Halloween ở một thành phố nọ, mọi người đang nô nức chuẩn bị những màn hóa trang đặc sắc và rùng rợn, những tưởng sẽ có một đêm tràn ngập niềm vui và hạnh phúc thì nỗi kinh hoàng ập đến… Mọi người dần phát hiện ra rằng, từ một lỗ đen bí ẩn bất chợt xuất hiện ra những con quái vật đáng sợ, nào là zombie, jack đồ tể cho tới cả khủng long bạo chú. Để bảo vệ tính mạng của mình và những người thân yêu, dân làng đã cùng nhau lập nên các tòa tháp để phong tỏa và tấn công bọn quái vật đang trên đường đi vào ngôi làng của mình. Nhưng để xây dựng những toàn tháp ấy cũng cần một cái giá của nó. Bạn hãy hóa thân vào người dân làng, với sự thông minh và tính toán hợp lý để xây dựng nên một phòng tuyến vững chắc bảo vệ ngôi làng…

III. Các bước tìm hiểu và thực hiện

1) Xây dựng map
   a) Ý tưởng ban đầu là dùng những ô gạch để ghép thủ công lại thình một map
   b) Sau đó, dùng vòng lặp để tạo những ô gạch đều nhau
   c) Phân loại ô gạch để có nhiều màu sắc và phù hợp với vị trí (đường đi, nơi đặt tower, nơi bắt đầu, nơi kết thúc)
   d) Cuối cùng, chuyển sang **load map từ file text số nguyên** với mỗi số là một loại gạch xếp thành một map
2) Camera: Di chuyển camera trong một map rộng lớn. Dùng phím di chuyển để lướt qua map sao cho mượt, tuy nhiên không vượt quá giới hạn của map.
3) Cấu trúc lưu trữ vị trí các ô gạch:
   a) Struct Point (x, y) thể hiện vị trí của ô gạch
   b) Dùng Dictionary để mapping từng vị trí tới các ô gạch cụ thể
   c) Xác định vị trí cũng như sprite cho vị trí hố đen xuất hiện quái vật và vị trí kết thúc (portal)
4) Quản lý truy cập instance của class
   a) Kiểu dữ liệu khái quát: Generic, để khái quát hóa các class
   b) Design pattern: **Singleton**, để lấy một instance duy nhất

5) Xây dựng và thao tác người chơi với các tháp (tower): Có 4 tháp với các mức giá khác nhau.
   a) Tìm kiếm sprite cho mỗi loại tháp
   b) Click chuột để đặt tháp lên map
   c) Xây dựng canvas để thể hiện trên giao diện hình ảnh các loại tháp
   d) Xử lý sự kiện (event) click vào các button tháp trên canvas. Click vào loại tháp nào sẽ đọc sprite của tháp đó
   e) Hover icon của tháp theo con trỏ chuột khi click vào button tháp
   f) Hủy chọn tháp (bằng chuột phải)
   g) Mỗi loại tháp ứng với một số tiền, cần quản lý tiền của người chơi, tiến hành trừ tiền khi mua tháp
6) Thuật toán di chuyển của enemies (tránh vật cản là các toà tháp)
   a) Tìm hiểu thuật toán **Astar**
   b) Xây dựng cấu trúc Node đại diện cho mỗi ô gạch
   c) Thay vì gán vị trí bắt đầu và kết thúc trong code. Tiến hành dựng sự kiện click chuột phải để tạm gán vị trí bắt đầu và kết thúc để debug
   d) Tiến hành dùng phím Space để tạm coi như là tín hiệu bắt đầu tìm đường đi ngắn nhất
   e) Trực tiếp debug trên game bằng cách tô màu cũng như thể hiện hướng đi và các hệ số f, g, h lên thẳng trên mỗi ô gạch
7) Sinh enemies (spawn) cho mỗi màn (wave)
   a) Tìm sprite cho enemies. Có 4 loại enemies: zombie nam, zombie nữ, jack đồ tể, khủng long bạo chúa. Tạo button “Next wave” bắt sự kiện để in ra random enemies tại vị trí hố đen xuất phát
   b) Tạo hiệu ứng xuất hiện cho enemies khi từ hố đen xuất phát
   c) Kết hợp sprite của enemies và thuật toán Astar để tiến hành di chuyển theo đường đi ngắn nhất tránh vật cản
   d) Tạo animation cho từng loại enemies
   e) Tạo hiệu ứng biến mất cho enemies khi đi đến vị trí kết thúc
   f) Tạo hiệu ứng cho các hố đen khi enemies chạm tới
   g) Tìm cách để tái sử dụng lại object enemies cho tạo ra thay vì cứ mỗi enemies tạo là một object: **Object Pool**
   h) Tiến hành thiết kế màn chơi, mỗi khi vượt qua một màn chơi thì số lượng enemies cũng tăng thêm. Cũng như chỉ cho phép người chơi mua tháp trước mỗi màn (wave) bắt đầu, trong lúc màn chơi đang diễn ra thì không được phép can thiệp để mua thêm tháp
8) Thiết kế game UI
   a) Số mạng của người chơi
   b) Menu game over (Restart, Quit)
   c) Logic khi restart game là rest lại mạng của người chơi, màn chơi và số tiền

d) Logic quit game đơn giản là close app

9) Hiệu ứng, logic xử lý cho các loại tháp và đạn
   a) Vẽ cho tháp một phạm vi/tầm bắn (range) thể hiện bằng vòng tròn màu xanh da trời
   b) Mỗi range của tháp sẽ là có một **collider2D**, mỗi enemies sẽ có một tag định danh để bắt được sự kiện khi enemies đi vào tầm bắn. Mỗi tower sẽ có một **Queue** để quản lý enemies khi chúng đi vào tầm bắn
   c) Tiếp theo, bắt đầu thiết kế đạn cho mỗi loại tháp, đạn sẽ được phóng ra từ tâm của tháp
   d) Tìm hàm để di chuyển đạn từ vị trí tháp tới dính vào vị trí enemies hiện thời đang di chuyển tới: **MoveTowards**. Vì các loại đạn cũng là một object nên cũng được quản lý bởi Object Pool để tái sử dụng và create/release dễ dàng.
   e) Đạn bắn ra từ tháp trong khi enemies đang di chuyển nên trong lúc đạn di chuyển trên đường đạn sẽ có một độ xoay (rotation). Dùng hàm **Quaternion.AngleAxis** để tạo ra một độ xoay nhất định
10) Thiết kế thanh máu (health bar) cho enemies
   a) Gắn thanh máu vào enemies
   b) Xử lý logic trừ máu khi enemies bị đạn từ tháp bắn trúng
   c) Hiệu ứng đạng bắn trúng enemies
   d) Hiệu ứng death cho enemies khi hết máu
11) Tháp có damage, proc (tỉ lệ phần trăm để xuất hiện hiệu ứng), debuff duration (thời gian để hiệu ứng tồn tại trên enemies): Sử dụng **Inheritance**
   a) Big Sand Tower: giá 2$, damage 5, proc 10% xác suất xảy ra hiệu ứng gây choáng trong một khoảng thời gian 1s
   b) Circle Sand Tower: giá 2$, damage 5, proc 10% xác suất hiệu ứng làm chậm enemies với một slowing factor là 50% và duy trì 5s
   c) Fire Tower: giá 4$, damage 5, proc 10% xác suất xảy ra hiệu ứng thiêu đốt duy trì 10s với mỗi 1s (tick time) gây 5 damage (tick damage)
   d) Stone Tower: giá 5$, damage 10, proc 10% xác suất xảy ra hiệu ứng duy trì trong 5s (debuff duration), sau mỗi 5s (tick time) nếu hiệu ứng vẫn còn duy trì thì để lại một cái bẫy trên đường đi, bẫy gây sát thương là 5 (slash damage) khi bị đạp trúng rồi biến mất
12) Hiện thông số của các tháp, bán và nâng cấp tháp
   a) Bán: khi bán tháp sẽ mất một nửa giá trị của tháp
   b) Nâng cấp: Với mỗi loại tháp, lưu sẵn mảng thông số nâng cấp gồm: giá tiền, damage, proc, debuff duration
13) Giao diện game: Pause game, Resume game, Restart game, Option (Music volume, Sound volume), Quit game

IV. Tính năng

1) Load map ô gạch động từ text file
2) Sử dụng thuật toán Astar để tìm đường đi từ điểm đầu đến điểm đích
3) Tạo hiệu ứng kỹ năng cho từng tháp
4) Health bar cho mỗi enemies
5) Mỗi loại tháp có các chỉ số khác nhau (damage, proc, …)
6) Nâng cấp tháp
7) Hiển thị chú thích cho mỗi loại tháp trong game
8) Điều kiện thua game, trừ máu, tăng tiền
9) Restart game
10) Camera di chuyển trong vùng không gian của map
11) Mua và bán tháp, đảm bảo không cho đặt tháp cho hết đường đi của enemies
12) Sinh ra enemies cho mỗi màn chơi
13) Hiệu ứng cho tháp và enemies dựa trên action hiện tại (di chuyển, ăn đạn, chết, bắn đạn, …)
14) Có hỗ trợ main menu, ingame menu và option menu
15) Thêm âm thanh và nhạc nền cho game
16) Có thể điều chỉnh âm lượng từ menu

V. Kỹ thuật và kiến trúc

1) Load map từ file text
   a) File text gồm các số nguyên được định dạng trước
   b) Đọc file bằng hàm readLevelText (LevelManager.cs)
   c) Dùng vòng lặp để duyện và thay thế mỗi số nguyên thành các loại ô gạch tương ứng

2) Camera Movement:
   a) Input: WASD. Di chuyển camera theo hướng

```
if (Input.GetKey(KeyCode.W))
{
    transform.Translate(Vector3.up * cameraSpeed * Time.deltaTime);
}
if (Input.GetKey(KeyCode.A))
{
    transform.Translate(Vector3.left * cameraSpeed * Time.deltaTime);
}
if (Input.GetKey(KeyCode.D))
{
    transform.Translate(Vector3.right * cameraSpeed * Time.deltaTime);
}
if (Input.GetKey(KeyCode.S))
{
    transform.Translate(Vector3.down * cameraSpeed * Time.deltaTime);
}
```

   b) Đảm bảo luôn trả về giá trị nằm trong khoảng max và min

```
transform.position = new Vector3(Mathf.Clamp(transform.position.x, 0, xMax), Mathf.Clamp(transform.position.y, yMin, 0),-10);
```

3) Singleton Design Pattern: thao tác được với instance duy nhất của một class

```
public abstract class Singleton<T> : MonoBehaviour where T : MonoBehaviour
{
    private static T instance;

    public static T Instance
    {
        get
        {
            if (instance == null)
            {
                instance = FindObjectOfType<T>();
            }
            return instance;
        }
    }
}
```

4) Thuật toán Astar: Dùng openList và closedList; score: F, G, H

```
public void CalcValues(Node parent, Node goal, int gCost)
{
    this.Parent = parent;
    this.G = parent.G + gCost;
    this.H = ((Math.Abs(GridPosition.X - goal.GridPosition.X)) + (Math.Abs(goal.GridPosition.Y - GridPosition.Y))) * 10;

    this.F = G + H;
}
```

5) Object Pool: tái sử dụng lại các object thay vì tạo mới: enemies, đạn …

```
public class ObjectPool : MonoBehaviour
{
    [SerializeField]
    private GameObject[] objectPrefabs;

    private List<GameObject> pooledObjects = new List<GameObject>();

    public GameObject GetObject(string type)
    {
        foreach(GameObject go in pooledObjects)
        {
            if(go.name == type && !go.activeInHierarchy)
            {
                go.SetActive(true);
                return go;
            }
        }

        for(int i = 0; i< objectPrefabs.Length; i++)
        {
            if(objectPrefabs[i].name == type)
            {
                GameObject newObject = Instantiate(objectPrefabs[i]);
                pooledObjects.Add(newObject);
                newObject.name = type;
                return newObject;
            }
        }
        return null;
    }

    public void ReleaseObject(GameObject gameObject)
    {
        gameObject.SetActive(false);
    }
}
```

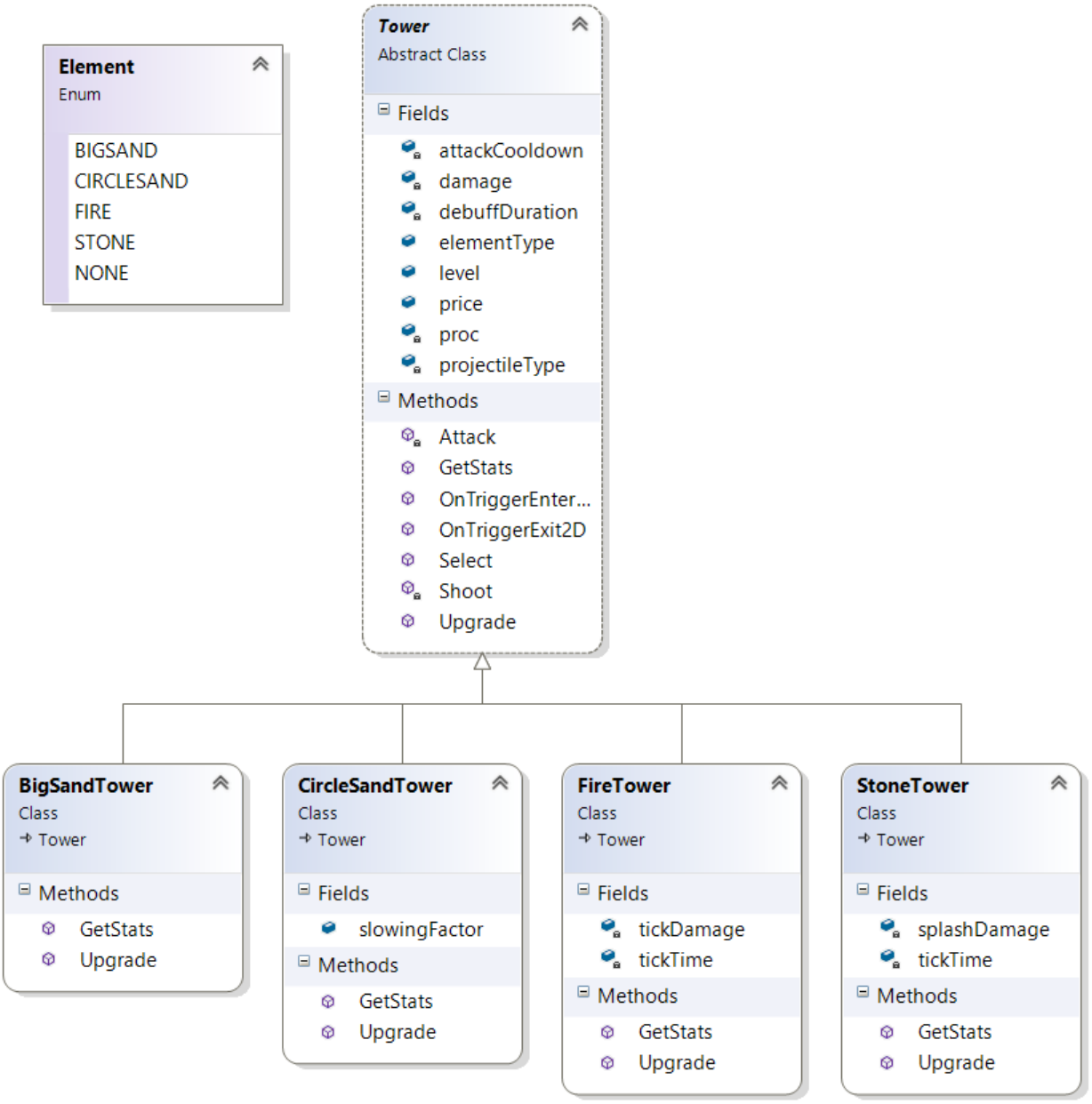
a) objectPrefabs: 4 loại enemies và 4 loại đạn (projectile). Tức là ta sẽ tạo một pool cho 4 object trên tái sử dụng
b) Hàm GetObject: Nếu object có nằm trong pool thì SetActive bằng true rồi trả về object đó. Nếu không thì xem object có thuộc objectPrefabs hay không, nếu thuộc thì add vào pool để lần sau tái sử dụng
c) Hàm ReleaseObject đơn giản là SetActive cho object về false

6) Đạn bắn ra từ tháp di chuyển tới mục tiêu enemies trong phạm vi tầm bắn và có một độ xoay nhất định

```
private void MoveToTarget()
{
    if (target != null && target.IsActive)
    {
        transform.position = Vector3.MoveTowards(transform.position, target.transform.position, Time.deltaTime * parent.ProjectileSpeed);

        Vector2 dir = target.transform.position - transform.position;

        float angle = Mathf.Atan2(dir.y, dir.x) * Mathf.Rad2Deg;

        transform.rotation = Quaternion.AngleAxis(angle, Vector3.forward);
    }
    else if (!target.IsActive)
    {
        GameManager.Instance.Pool.ReleaseObject(gameObject);
    }
}
```

a) Di chuyển của đạn sẽ tuân theo hàm MoveTowards từ vị trí bắn đạn tới mục tiêu với một tốc độ cho trước
b) Tính vector “dir” từ vị trí bắn đến mục tiêu
c) Tính góc radian bằng của tan (dir.y/dir.x) sau đó đổi ra độ
d) Xác định độ xoay để xoay “angle” độ quanh trục Vector3.forward (0, 0, 1)

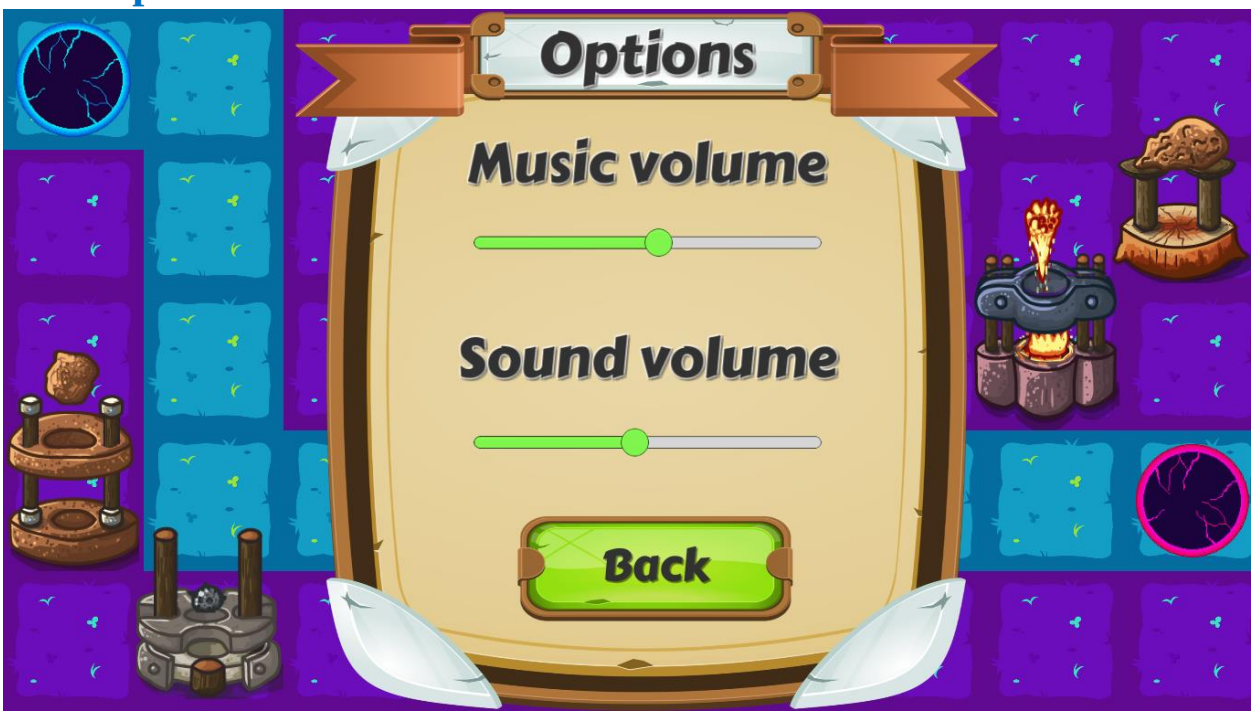
7) Tính kế thừa trong game: Như đã mô tả ở Phần III.11). Mỗi loại tháp sẽ có một đặc tính, hiệu ứng, chỉ số, cách nâng cấp khác nhau.



VI. Screenshots

1. Main menu

2. Option menu

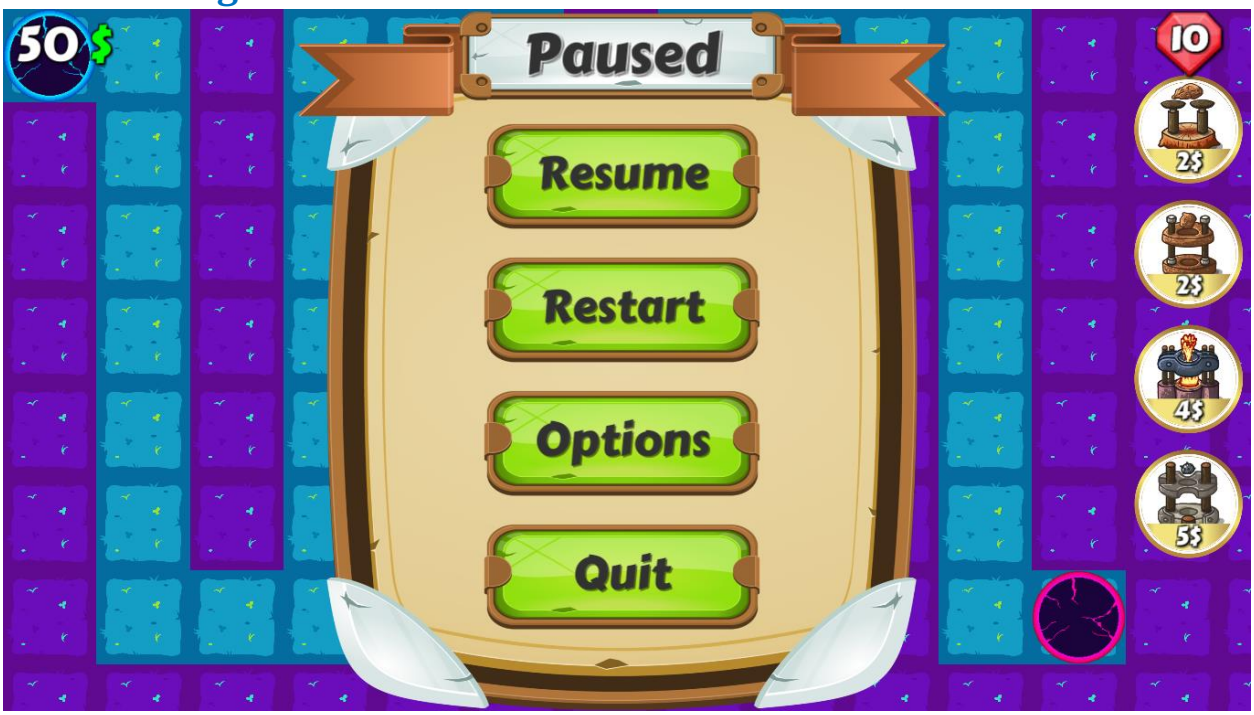


3. Hover tower button

4. Sell and upgrade tower

5. Hit enemies

6. Pause game



7. Game over

VII. Demo

1) Link youtube: https://www.youtube.com/watch?v=svZTxTA7hKw

2) Link github: https://github.com/1612842/TowerDefense

VIII. Nguồn tham khảo

- Sprite:
  - Khủng long: https://www.gameart2d.com/free-dino-sprites.html
  - Zombie nam, nữ: https://www.gameart2d.com/the-zombies-free-sprites.html
  - Jack đồ tể: https://www.gameart2d.com/jack-o-lantern-free-sprites.html
  - Towers: https://free-game-assets.itch.io/free-stone-tower-defense-game-art
  - Portals: https://www.spriters-resource.com/arcade/dariusgaiden/sheet/99743/
  - Tiles: https://inscope.itch.io/2d-tilemap
- Audio:
  - SFX sound: https://www.freesfx.co.uk/sfx/splaterf.mp3

    - Music in game: https://www.bensound.com/bensound-music/bensound-epic.mp3
    - Music in menu: https://www.nhaccuatui.com/bai-hat/thriller-michael-jackson.sciP4GxVOQXd.html
- Poster:
    - https://www.freepik.com/free-vector/realistic-halloween-pumpkin-banners_5616238.htm#page=3&query=halloween&position=26
    - https://www.freepik.com/free-vector/halloween-silhouette-collection_5567248.htm
    - https://www.freepik.com/free-vector/flyer-halloween-party-brown-tones_1347731.htm#page=1&query=spooky-halloween-party-poster-with-flat-design&position=15
    - https://seeklogo.net/html5-eps-ai-crd-file5899-download.html
    - https://seeklogo.net/android-robot-ai-eps-crd-file1606-download.html
    - https://seeklogo.net/tag/microsoft-windows

-Hết-