**BÁO CÁO MÔN HỌC**

**LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG  
Đề tài: Application**

**Giảng viên: Ths. Phạm Thị Kim Ngoan**

**Danh sách thành viên:**

**1. Dương Ngọc Thoại**

**2. Phan Thanh Hà**

**3. Phan Trương Duy Luận**

**4. Nguyễn Văn Trí**

**5. Lê Văn Thật**

**Lớp : 60.CNTT-3**

**MỤC LỤC**

[**NỘI DUNG** 2](#_Toc25962938)

[**I. Nêu bài toán:** 2](#_Toc25962939)

[1. Tạo game 2](#_Toc25962940)

[2. Yêu cầu thực hiện 2](#_Toc25962941)

[**II. Cách giải quyết:** 3](#_Toc25962942)

[1.Hướng giải quyết 3](#_Toc25962943)

[2.Quy trình thiết kế ứng dụng 3](#_Toc25962944)

[B1: Thiết kế giao diện cơ bản 3](#_Toc25962945)

[B2: Lập trình tính năng xử lí cơ bản 11](#_Toc25962946)

[B3: Lập trình tính năng xử lí bổ sung 14](#_Toc25962947)

[B4: Hoàn tất thiết kế giao diện 14](#_Toc25962948)

[**III. Cài đặt thực hiện:** 14](#_Toc25962949)

[3.1 Mã nguồn 14](#_Toc25962950)

[3.2 Giao diện 15](#_Toc25962951)

[**IV. Tài liệu :** 16](#_Toc25962952)

[**V. Phân công:** 16](#_Toc25962953)

[1. Lựa chọn và phân tích đề tài 16](#_Toc25962954)

[2. Ý tưởng về thiết kế giao diện, chức năng 16](#_Toc25962955)

[3. Giải quyết bài toán 17](#_Toc25962956)

[4. Tiến hành code mã nguồn 17](#_Toc25962957)

[5. Tiến hành kiểm thử 17](#_Toc25962958)

[6. Viết báo cáo và tổng hợp 17](#_Toc25962959)

# 

# **NỘI DUNG**

# **I. Nêu bài toán:**

## 1. Tạo game : Đuổi hình bắt chữ trên giao diện Winform:

Game đuổi hình bắt chữ là gì?

Ý tưởng: là trò chơi mô phỏng theo gameshow nổi tiếng trên truyền hình Đuổi hình bắt chữ, màn hình game sẽ hiện lên một hình ảnh cho bạn đoán, hãy vận dụng khả năng tưởng tượng của bạn để đoán xem liệu đó là câu gì rồi đưa ra câu trả lời phía bên dưới, nếu đúng bạn sẽ qua câu tiếp theo. Nếu bạn trả lời đúng hết tất cả câu hỏi bạn sẽ là người chiến thắng.

## 2. Yêu cầu thực hiện:

- Tạo giao diện chính bao gồm: Hình ảnh gợi ý, bảng các kí tự gợi ý, bảng kí tự trả lời đáp án.

- Tạo ra các kí tự chữ cái, khi click vào kí tự thì lập tức kí tự xuất hiện trong từng ô đáp án theo thứ tự từ trái sang phải và khi click ngược lại tại từng ô đáp án kí tự trở về với bảng gợi ý

* Khi đã lấp đầy các kí tự vào ô đáp án thì sẽ bắt đầu kiểm tra đúng/ sai nếu kết quả đúng thì thông báo “đúng rồi” vào cho qua câu hỏi tiếp theo, nếu sai thì thông báo “ sai rồi” và bạn phải kiểm tra và sửa lại đáp án phù hợp cho tới khi đáp án đúng mới được qua câu còn
* Nếu bạn trả lời câu hỏi cuối cùng đúng thì xuất ra dòng thông báo “ Bạn chiến thắng”.

# **II. Cách giải quyết:**

## 1.Hướng giải quyết

- Lựa chọn lập trình trên Winform của C# ( Visual studio): Tiết kiệm thời gian, dễ thao tác.

- Dựa vào yêu cầu thực hiện bài toán, ta xử lí như sau:

+ Mỗi câu hỏi là 1 tấm hình khác nhau

=> Có sẵn danh sách các tấm hình và chúng ta đọc ngẫu nhiên các tấm hình, cứ thế trả lời đúng thì mới qua tấm mới

+ Luôn cố định tầm 14 chữ cái đáp án gợi ý, còn lại tùy theo tấm hình mà đáp án lưu sẵn sẽ xuất hiện tương ứng bấy nhiêu ô đáp án

=> vì thế khi tạo ra game này với 1 tấm hình đưa vào chúng ta đã phải có sẵn đáp án và dựa theo đáp án đó mà chúng ta biết nó có bao nhiêu ký tự => Chúng ta sẽ phát sinh ra bấy nhiêu ô dữ liệu đáp án tương ứng (Text Box)

Vd: Đáp án là CADAO => phát sinh ra 5 ô TextBox

Đáp án là CANBANG => phát sinh ra 8 ô TextBox

+ Mỗi ô TextBox cách nhau 1 khoảng cách => Vị trí (Location) kết hợp với Size của TextBox

+ Số chữ cái đáp án luôn chắc chắn có nằm trong danh sách chữ cái gợi ý, danh sách chữ cái gợi ý sẽ Random ngẫu nhiên không theo thứ tự nhưng luôn đảm bảo có đủ số chữ cái đáp án

Kiểm tra kết quả khi: Các ô đáp án được lấp đầy

==========================

## 2.Quy trình thiết kế ứng dụng

B1. Thiết kế giao diện cơ bản

B2. Lập trình tính năng xử lý cơ bản

B3. Lập trình tính năng xử lý bổ sung

B4. Hoàn tất thiết kế giao diện

==========================

Quy trình thiết kế game:

### B1: Thiết kế giao diện cơ bản:

1. Hiện thị ngẫu nhiên các hình ảnh:

* Tấm hình Random ra được là nhờ 1 biến chỉ số (tạm đặt là index) truyền vào. Vậy làm thế nào để việc Random không hiển thị lại tấm hình đã xuất hiện?

→ Chỉ cần index không trùng nhau thì tấm hình sẽ không trùng nhau.

* Mỗi lần Random ra kết quả index sẽ so sánh với các index đã Random ra trước đó (được lưu vào một mảng những index đã được sử dụng), nếu trùng sẽ Random lại, nếu không trùng thì lấy index đó và lưu index đó vào mảng những index đã được sử dụng.

**Code:**

listIndexDaDung = new int[listHinhAnh.Length];

//random ra 1 tấm hình bất kì

Random rd = new Random();

index = rd.Next(0, listHinhAnh.Length);

indexdadung = 0;

listIndexDaDung[indexdadung] = index;

1. Sinh ra các ô đáp án:

* Bản thân khi tạo ra game này với 1 tấm hình đưa vào chúng ta đã phải có sẵn đáp án và dựa theo đáp án mà chúng ta biết nó có bao nhiêu ký tự

→ Chúng ta sẽ phát sinh ra bấy nhiêu ô đáp án tương ứng (TextBox).

* Vị trí xuất hiện các ô đáp án: Location(x ,y)
  + Sinh các ô đáp án là 1 mảng các TextBox[0, n – 1]
  + Phân tích:

x của Textbox[0] = x của tấm hình.

y của TextBox[0] = y của tấm hình + Height của tấm hình + 1 khoảng cách.

x của TextBox[1] = x của TextBox[0] + Width của TextBox[0] + 1 khoảng cách.

y của TextBox[1] = y của TextBox[0].

x của TextBox[2] = x của TextBox[1] + Width của TextBox[1] + 1 khoảng cách.

y của TextBox[2] = của TextBox[0].

...

x của TextBox[i] = x của TextBox[i - 1] + Width của TextBox[i - 1] + 1 khoảng cách.

y của TextBox[i] = y của TextBox[0].

→ Vậy ta cần sinh ra TextBox[0] trước, từ đó dùng vòng lặp để sinh ra các TextBox còn lại.

**Code:**

int soluongtextbox = listDapAn[index].Length;

txtDapAn = new TextBox[soluongtextbox]; // số lượng text box = n

// phát sinh ô TextBox[0]

txtDapAn[0] = new TextBox();

txtDapAn[0].Location = new Point(ptrHinhAnh.Location.X,ptrHinhAnh.Location.Y + ptrHinhAnh.Size.Height + 10);

// Sử dụng vòng lặp sinh ra các ô TextBox còn lại

for (int i = 1; i < soluongtextbox; i++)

{

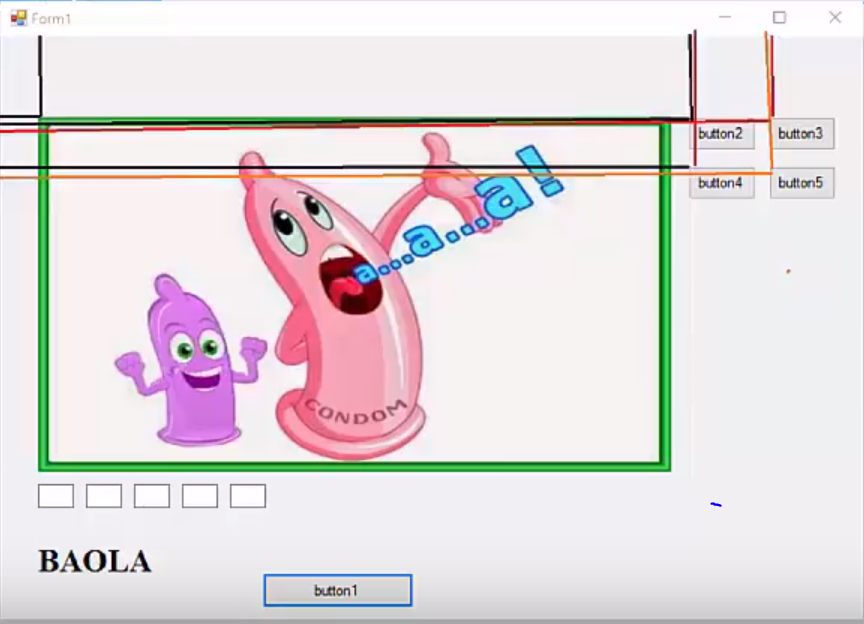
txtDapAn[i] = new TextBox();

txtDapAn[i].Location = new Point(txtDapAn[i - 1].Location.X + txtDapAn[i - 1].Size.Width + 10, txtDapAn[0].Location.Y);

}

1. Sinh ra các ô gợi ý đáp án:

* Cũng giống như việc sinh ra các ô đáp án và các ô gợi ý đáp án được sinh ra phải nhiều hơn số ô đáp án. Số ô gợi ý đáp án được sinh ra có thể là một số cụ thể, cố định hoặc theo một nguyên tắc nào đó.
* Ở bài này, nhóm mình thực hiện theo nguyên tắc: Nếu đáp án có x ký tự thì tạo ra (x \* 2) các ô gợi ý đáp án. Ngoài trừ x ký tự đáp án thì x còn lại sẽ random ra trong bảng chữ cái sau đó tất cả xáo trộn lại 1 lần cuối rồi hiển thị ra danh sách gợi ý đáp án.
* Vị trí xuất hiện các ô gợi ý đáp án: Location(x, y)
  + Sinh các ô gợi ý đáp án là 1 mảng các Button[0, n – 1], chia làm 2 cột: 1 cột các Button[chẵn] và 1 cột các Button[lẻ]
  + Phân tích:



Button[0]

X = Location.X của Picture + Width của Picture + 1 khoảng cách

Y = Location.Y của Picture

Button[1]

X = Location.X của Button[0] + Width của Button[0] + 1 khoảng cách

Y = Location.Y của Button[0]

Button[2]

X = Location.X của Button[0]

Y = Location.Y của Button[0] + 1 khoảng cách.

Button[3]

X = Location.X của Button[1]

Y = Location.Y của Button[1] + 1 khoảng cách.

...

Button[i]

X = Location.X của Button[i – 2]

Y = Location.Y của Button[i – 2] + 1 khoảng cách.

→ Vậy ta cần sinh ra Button[0] và Button[1] trước, từ đó dùng vòng lặp để sinh ra các Button còn lại.

**Code:**

listGoiYDapAn = new Button[txtDapAn.Length \* 2];

// phát sinh ô Button[0]

listGoiYDapAn[0] = new Button();

listGoiYDapAn[0].Location = new Point(ptrHinhAnh.Location.X + ptrHinhAnh.Size.Width + 30, ptrHinhAnh.Location.Y);

// phát sinh ô Button[1]

listGoiYDapAn[1] = new Button();

listGoiYDapAn[1].Location = new Point(listGoiYDapAn[0].Location.X + listGoiYDapAn[0].Size.Width + 30, listGoiYDapAn[0].Location.Y);

// Sử dụng vòng lặp sinh ra các ô Button còn lại

for (int i = 2; i < listGoiYDapAn.Length; i++)

{

listGoiYDapAn[i] = new Button();

listGoiYDapAn[i].Location = new Point(listGoiYDapAn[i - 2].Location.X, listGoiYDapAn[i - 2].Location.Y + 30);

}

* Sau khi chúng ta đã có số lượng ô gợi ý đáp án tương ứng thì sẽ làm việc sau đây: Tạo ra danh sách các chữ cái lấp đầy lên các ô sao cho luôn đảm bảo có đủ số chữ cái đáp án ở trong đó và tất cả xáo trộn với nhau.
  + listGoiYDapAn có n phần tử (n = x \* 2) với x chính là số lượng ký tự đáp án:

50% sẽ là các ký tự đáp án.

50% sẽ là các ký tự ngẫu nhiên phát sinh trộn lẫn lại với nhau.

* + Cách trộn lẫn: Chúng ta có thể dựa theo một quy tắc nào đó để trộn lẫn. Ví dị như: kí tự đáp án nằm ở vị trí chẵn; kí tự đáp án nằm theo chuỗi vị trí 1 4 5 3 2... Nhưng việc trộn lẫn theo một quy tắc như vậy sẽ làm game mất hay, bởi vì khi người chơi chơi nhiều sẽ phát hiện ra được quy luật đó. Vậy cách trộn lẫn tốt nhất để game hay và thú vị thì đó là trộn lẫn không theo một quy tắc nào cả (mọi thứ phụ thuộc hoàn toàn vào RANDOM).
  + Tạo 1 mảng 1 chiều lưu trữ đủ các ký tự đáp án và phát sinh ngẫu nhiên các ký tự còn lại, có chỉ số index từ 0 đến n – 1 (với n là số phần tử của mảng ⬄số lượng ký tự gợi ý đáp án). Sau đó Random 1 chỉ số index bất kỳ trong danh sách các chỉ số index của mảng và kiểm tra không bị trùng nhau rồi đọc lên Button theo tuần tự.

**Code:**

// Đưa vào các ký tự đáp án

char[] DanhSachKyTuDapAnGoiY = new char[listGoiYDapAn.Length];

for(int i = 0; i < DanhSachKyTuDapAnGoiY.Length / 2; i++)

{

char kytu = listDapAn[index][i];

// Xử lý luôn vấn đề nếu chữ cái đáp án là thường thì sẽ hiển thị ra hoa

if(Char.IsLower(kytu))

{

DanhSachKyTuDapAnGoiY[i] = (char)(kytu - 32); // A: 65 , a: 97

}

else

{

DanhSachKyTuDapAnGoiY[i] = kytu;

}

}

// Phát sinh ngẫu nhiên ra các ký tự còn lại

Random rd = new Random();

// Phạm vi random trong bảng chữ cái viết hoa (MÃ ASCII) // A -> Z là : 65 -> 90

// trừ những chữ cái như F , J , Z , W

for (int i = DanhSachKyTuDapAnGoiY.Length / 2; i < DanhSachKyTuDapAnGoiY.Length; i++)

{

int kt = rd.Next(65, 91);

while (kt == 70 || kt == 74 || kt == 90 || kt == 87)

{

kt = rd.Next(65, 91); // Ép kiểu char lấy ra ký tự có mã ASCII tương ứng

}

DanhSachKyTuDapAnGoiY[i] = (char)kt;

}

### B2: Lập trình tính năng xử lí cơ bản:

**Code:**

// Trộn lẫn và đọc lên Button các ký tự gợi ý đáp án tương ứng

int[] mangindexdadung = new int[listGoiYDapAn.Length];

int soluong = 0;

for(int i = 0; i < listGoiYDapAn.Length; i++)

{

bool Check;

do

{

int idx = rd.Next(0, listGoiYDapAn.Length);

// Kiểm tra bị trùng hay không ?

Check = true;

for (int j = 0; j < soluong; j++)

{

if (idx == mangindexdadung[j])

{

Check = false;

break;

}

}

if (Check == true)

{

// Hiển thị ký tự lên nút Button

listGoiYDapAn[i].Text = DanhSachKyTuDapAnGoiY[idx].ToString();

// Cập nhập idx vào danh sách các index đã dùng của danh sách các ký tự gợi ý đáp án

mangindexdadung[soluong++] = idx;

}

} while (Check == false);

}

1. Điền đáp án: Chọn 1 ngẫu nhiên 1 ô gợi ý đáp án, thì tự động nó kiểm tra ô đầu tiên từ trái sang phải nếu chưa có đáp án thì sẽ điền vào và nút button đó sẽ mất đi.
2. Chỉnh sửa đáp án: Khi người chơi muốn bỏ đi 1 ký tự trong ô đáp án bất kỳ thì sẽ Click vào ô đáp án bất kỳ đó thì ký tự này sẽ mất đi và nút Button có ký tự tương ứng đã bị ẩn đi trước đó nay sẽ hiện lên lại.

**Code:**

else if(sender is TextBox)

{

// Lấy Text đó đi vào danh sách các Button đã bị ẩn đi để kiểm tra xem Button nào có Text tương ứng thì cho hiển thị lên lại

foreach(Button btn in listGoiYDapAn)

{

// Kiểm tra tình trạng nút nhấn đang bị ẩn

if(btn.Visible == false && btn.Text == ((TextBox)sender).Text)

{

btn.Visible = true; // Hiện lên lại

break; // Thoát khỏi vòng lặp vì mỗi lần là 1 textbox

}

}

}

**Code:**

if (sender is Button)

{

//Điền Text từ Button vào ô TextBox rỗng theo chiều từ trái sang phải ( ô đầu tiên rỗng)

for (int i = 0; i < txtDapAn.Length; i++)

{

if (txtDapAn[i].Text == "") // Gặp ô đầu tiên rỗng thì điền vào

{

txtDapAn[i].Text = ((Button)sender).Text; // Điền Vào

((Button)sender).Visible = false; // ẩn nút Button đi

break; // thoát khỏi vòng lặp

}

}

}

1. Kiểm tra đáp án: Khi lấp đầy được hết các ô đáp án nó sẽ kiểm tra đúng/sai rồi sẽ ra kết quả
2. Chuyển câu kế tiếp: Sau khi kiểm tra tính đúng/sai nếu câu trả lời sai thì thông báo “Sai rồi” còn nếu câu trả lời đúng thì thông báo đúng rồi đồng thời tự động chuyển sang câu tiếp theo.

**Code:**

// Kiểm tra tình trạng các ô TextBox đã được lấp đầy hay chưa để từ đó đi kiểm tra đúng/sai

bool Check = false;

foreach (TextBox txt in txtDapAn)

{

if (txt.Text == "")

{

Check = true;

break; // Đã phát hiện 1 ô TextBox rỗng => chưa đủ điều kiện để tính thắng/thua

}

}

if (Check == false)

{

// Tất cả ô TextBox đã được điền đầy đủ => Kiểm tra đúng/sai

string s = "";

// Duyệt qua tất cả các ô TextBox chứa các ký tự đáp án người chơi đã chọn để nối lại thành 1 chuỗi rồi đem đi so sánh

for (int i = 0; i < txtDapAn.Length; i++)

{

s = s + txtDapAn[i].Text;

}

// So sánh s với đáp án để biết đúng/sai. Lưu ý: Không phân biệt hoa thường

if (s.ToUpper() == listDapAn[index].ToUpper()) // Đổi hết sang hoa và so sánh với nhau

{

MessageBox.Show("Đúng rồi!");

ChuyenQuaCauTiepTheo();// d. Chuyển câu kế tiếp: xem thêm trong phương thức ChuyenQuaCauTiepTheo() trong source code

}

else

{

MessageBox.Show("Sai Rồi!");

}

}

### B3: Lập trình tính năng xử lí bổ sung: Phần này để phát triển thêm cho game có nhiều tính năng thú vị, hấp dẫn hơn. Ví dụ như:

* Luôn có điểm cho từng câu (điểm này có thể cố định hoặc tùy theo mức độ khó mỗi câu sẽ có mức điểm khác nhau) – Tùy theo điểm mà sẽ có phần quà tương ứng.

TH1: Điểm luôn cố định mỗi câu.

TH2: Điểm tùy theo câu hỏi (có câu 3 điểm, có câu 10 điểm).

TH3: Điểm tùy theo cột mốc (10 câu đầu là 10 điểm/câu, 10 câu tiếp theo là 20 điểm/câu).

* Chức năng gợi ý: Mở ra sẽ mất 1 số lượng điểm tương ứng để giúp chúng ta đi qua được câu hỏi này.
* Giống như game Ai là Triệu Phú: 50/50 (Mở 1 nửa các ô đáp án ở vị trí bất kỳ và ẩn đi các nút Button tương ứng có ký tự đã mở).
* Muốn game khó hơn: Cho thêm thời gian trả lời cho mỗi câu hỏi (Ví dụ: 5 phút) và có đồng hồ đếm ngược.
* Chức năng xếp hạng điểm → ghi tập tin.

### B4: Hoàn tất thiết kế giao diện:

* Canh chỉnh lại vị trí các thành phần hiển thị trên form để cho ra 1 giao diện đẹp.
* Thêm các hiệu ứng nhấn, giữ chuột vào các đối tượng.
* Chỉnh sửa phông chữ, màu chữ, cỡ chữ...
* Thêm một số hình ảnh trang trí.

# **III. Cài đặt thực hiện:**

## 3.1 Mã nguồn: 3 cửa sổ Form1.cs, Form1.Desiger.cs,Form1.cs\*

## 3.2 Giao diện :

• Màn hình chính (Chưa nhấn nút START)

• Sau khi nhấn nút START

# **IV. Tài liệu :**

- Kiến thức lập trình OOP with C# (Đã được học tại trường)

- Kiến thức chung về winform: Trang web tham khảo: Howkteam

<https://www.howkteam.vn/course/lap-trinh-winform-co-ban-27>

- Phân tích và hướng dẫn tự làm game: Đuổi hình bắt chữ

<https://www.youtube.com/watch?v=BsxEOu-ZnMU&list=PLq9VOPepajsBqkpGZQED4SmTyFVuWQRZM&index=40&t=0s>

# **V. Phân công:**

## 1. Lựa chọn và phân tích đề tài:

- Nguyễn Văn Trí

- Dương Ngọc Thoại

## 2. Ý tưởng về thiết kế giao diện, chức năng:

- Phan Trương Duy Luận

- Lê Văn Thật

## 3. Giải quyết bài toán:

- Dương Ngọc Thoại

- Lê Văn Thật

- Phan Trương Duy Luận

## 4. Tiến hành code mã nguồn:

- Nguyễn Văn Trí

- Phan Thanh Hà

## 5. Tiến hành kiểm thử: Cả nhóm

## 6. Viết báo cáo và tổng hợp:

- Dương Ngọc Thoại

- Nguyễn Văn Trí

- Phan Thanh Hà