MỤC LỤC

[MỤC LỤC 1](#_Toc30138191)

[**CÁC HÀM XỬ LÝ CHÍNH TRONG MÔ PHỎNG TRÊN C#** 2](#_Toc30138192)

[**1. Phần tử và các hàm có trong phần tử** 2](#_Toc30138193)

[**2. Các hàm tạo mảng** 6](#_Toc30138194)

[**3. Các hàm bắt đầu, hủy quá trình sắp xếp và các hàm hỗ trợ** 10](#_Toc30138195)

[**4. Các hàm di chuyển vị trí của Node và hoán vị hai Node** 13](#_Toc30138196)

[**5. Hàm tạm dừng quá trình sắp xếp** 14](#_Toc30138197)

[**6. Hàm chọn từng dòng code trong khung Code C/C++** 15](#_Toc30138198)

[***7.* Cập nhật trạng thái ý tưởng thuật toán** 15](#_Toc30138199)

[***8.* Các hàm sắp xếp mảng** 17](#_Toc30138200)

## **CÁC HÀM XỬ LÝ CHÍNH TRONG MÔ PHỎNG TRÊN C#**

### **1. Phần tử và các hàm có trong phần tử**

1. Đối tượng thao tác chính trong phần mềm là Node. Node là một UserControl gồm một Button chứa các phần tử: value (giá trị), vị trí hiện tại,… và một TextBox để người dung có thể nhập giá trị cho Node mỗi khi double-clicked

|  |
| --- |
| class Node : UserControl  {  private int value;  public static bool IsPause = false;  public int vitriHienTai;  public event EventHandler Capnhat;  public Node(int vitrihientai)  {  this.Width = this.Height = btn1.Width = btn1.Height = tbx1.Width = tbx1.Height = ThamSo.KichCoNode;  tbx1.Location = this.Location;  btn1.Location = this.Location;  tbx1.TextAlign = HorizontalAlignment.Center;  this.Size = new Size(ThamSo.KichCoNode, ThamSo.KichCoNode);  this.Anchor = (AnchorStyles.Top | AnchorStyles.Left);  btn1.BackColor = Color.Red;  tbx1.BackColor = Color.White;  tbx1.MaxLength = 2;  tbx1.AcceptsReturn = true;  btn1.Font = tbx1.Font = new Font("Consolas", ThamSo.KichCoNode / 4, FontStyle.Bold);  this.vitriHienTai = vitrihientai;  tbx1.LostFocus += Tbx1\_LostFocus;    }  } |

1. Để xử lý việc nhập giá trị bằng tay cho Node ta cần một số hàm chính sau:
   1. Hàm Tbx1\_LostFocus : Không cho người dùng nhập giá trị rỗng, khi TextBox nhập tay bị Lost-Focus thì tắt TextBox đi và lưu giá trị của TextBox vào trong Node.

|  |
| --- |
| private void Tbx1\_LostFocus(object sender, EventArgs e)  {  if (tbx1.Text=="")  {  tbx1.Text = "0";  }  else  {  this.Text = tbx1.Text;  this.Value = Int32.Parse(this.Text);  }  btn1.Visible = true;  btn1.BringToFront();  btn1.Focus();  if (Capnhat != null)  {  Capnhat(sender, e);  }  } |

* Hàm btn1\_MouseDown : Bắt sự kiện double-click cho TextBox để cập nhật giá trị cho Node.

|  |
| --- |
| private void btn1\_MouseDown(object sender, MouseEventArgs e)  {  if (e.Clicks == 2)  {  btn1.Visible = false;  tbx1.Text = btn1.Text;  tbx1.SelectionStart = tbx1.Text.Length;// dat con tro chuot "cusor" qua ben phai neu so da co  tbx1.BringToFront();  tbx1.Focus();    }  } |

* Hàm nhapTayTexbox\_KeyDown : nếu nhấn Enter thì tắt TextBox nhập tay và lưu giá trị vào Node.

|  |
| --- |
| private void tbx1\_KeyDown(object sender, KeyEventArgs e)  {  if (e.KeyCode == Keys.Enter)  {  btn1.Visible = true;  btn1.BringToFront();  btn1.Focus();  }  } |

* Hàm nhapTayTexbox\_KeyPress : chỉ cho phép nhập các phím là số khi nhập giá trị vào TextBox.

|  |
| --- |
| private void tbx1\_KeyPress(object sender, KeyPressEventArgs e)  {  if (!Char.IsDigit(e.KeyChar))  {  e.Handled = true;  }  } |

- Để xử lý di chuyển các Node lên, xuống, qua trái, qua phải ta cần các hàm sau:

* Hàm ChuyenXuong : chuyển Node đi xuống.

|  |
| --- |
| public void ChuyenXuong()  {  int y\_ViTriMoi = this.Location.Y+ ThamSo.DoCaoDiChuyen;  while (this.Location.Y < y\_ViTriMoi)  {  pauseStatus.WaitOne(Timeout.Infinite);  this.Location = new Point(this.Location.X, this.Location.Y + 1);  Thread.Sleep(ThamSo.ThoiGianDoi);  }  } |

* Hàm ChuyenLen : chuyển Node đi lên

|  |
| --- |
| public void ChuyenLen()  {  int y\_ViTriMoi = this.Location.Y - ThamSo.DoCaoDiChuyen;  while (this.Location.Y > y\_ViTriMoi)  {  pauseStatus.WaitOne(Timeout.Infinite);  this.Location = new Point(this.Location.X, this.Location.Y - 1);  Thread.Sleep(ThamSo.ThoiGianDoi);  }  } |

* Hàm ChuyenQua : chuyển Node đi qua trái hoặc qua phải tùy theo vị trí hiện tại của Node và vị trí mới mà Node chuyển đến.

|  |
| --- |
| public void ChuyenNgang(int vitriMoi)  {  int x\_vitririMoi;  if (vitriMoi > this.vitriHienTai)  {  x\_vitririMoi = this.Location.X + ((vitriMoi - vitriHienTai) \* (ThamSo.KhoangCachCacNode+ThamSo.KichCoNode));  while (this.Location.X < x\_vitririMoi)  {  pauseStatus.WaitOne(Timeout.Infinite);  Thread.Sleep(ThamSo.ThoiGianDoi);  this.Location = new Point(this.Location.X + 2, this.Location.Y);  }  }  else  {  x\_vitririMoi = this.Location.X - ((vitriHienTai - vitriMoi) \* (ThamSo.KhoangCachCacNode+ThamSo.KichCoNode));  while (this.Location.X > x\_vitririMoi)  {  pauseStatus.WaitOne(Timeout.Infinite);  Thread.Sleep(ThamSo.ThoiGianDoi);  this.Location = new Point(this.Location.X - 2, this.Location.Y);  }  }  } |

### **2. Các hàm tạo mảng**

- Để tạo một mảng các phần tử ta cần tạo ra một mảng các Node và lưu các Node lại bên trong một danh sách liên kết đơn để thuận tiện cho việc điều khiển

Một số các thuộc tính cần để tạo một mảng:

isRunning : cho biết quá trình sắp xếp có đang chạy hay không

DanhSachNode : chứa mảng các Node

labelSTTArr: chứa dãy số thứ tự bên dưới mảng

bienArr: Chứa label các biến có trong thuật toan

MangChuaSapXep: Chứa dãy số ban đầu

|  |
| --- |
| private bool isRunning;  public List<Node> DanhSachNode;  private List<Label> labelSTTArr = new List<Label>();  private Dictionary<string, Label> bienArr;  public List<int> MangChuaSapXep; |

- Các hàm tạo mảng:

1. Hàm VeNut: vẽ một Node

void VeNut()

{

DanhSachThamSo = new List<int>(SoLuongNode);

DanhSachNode = new List<Node>(SoLuongNode);

danhSachLabel = new List<Label>(SoLuongNode);

sapxepPanel.Controls.Clear(); // Xóa những btn cũ trên panel ở phiên làm việc

int temp = -SoLuongNode / 2;

int temp2 = ThamSo.KhoangCachCacNode / 2;

int temp3 = temp + 1;

Point newPoint;

for (int i = 0; i < SoLuongNode; i++)

{

Label tam = new Label() { Text = i.ToString() };

tam.Size = new Size(ThamSo.KichCoNode / 2, ThamSo.KichCoNode / 2);

tam.ForeColor = Color.Blue;

Node btn = new Node(i);

int Value = rank.Next(2, 100);

btn.Value = Value;

btn.Text = Value.ToString();

btn.BackColor = ThamSo.mauNen;

if (SoLuongNode % 2 != 0)

{

newPoint = new Point(sapxepPanel.Width / 2 - btn.Width / 2,sapxepPanel.Height / 2 - btn.Height / 2);

btn.Location = new Point(newPoint.X + temp \* btn.Width + temp \* ThamSo.KhoangCachCacNode, newPoint.Y);

temp++;

tam.Location = new Point(btn.Location.X+btn.Width/2 - tam.Width / 2, btn.Location.Y + 120);

}

else

{

if (i == SoLuongNode / 2)

{

temp2 = -temp2;

temp3 = 0;

}

newPoint = new Point(sapxepPanel.Width / 2 - btn.Width / 2 + ThamSo.KichCoNode / 2, sapxepPanel.Height / 2 - btn.Height / 2);

btn.Location = new Point(newPoint.X - temp2 + temp3 \* ThamSo.KhoangCachCacNode + temp \* btn.Width, newPoint.Y);// Node Xuất hiện giữa panel

tam.Location = new Point(btn.Location.X + btn.Width / 2-tam.Width/2, btn.Location.Y + 120);

temp++;

temp3++;

}

sapxepPanel.Controls.Add(tam);

sapxepPanel.Controls.Add(btn);

DanhSachNode.Add(btn);

DanhSachThamSo.Add(Value);

danhSachLabel.Add(tam);

btn.Capnhat += Btn\_Capnhat;

}

if (isRunning == false)

{

MangChuaSapXep = new List<int>(SoLuongNode);

for (int i = 0; i < SoLuongNode; i++)

{

MangChuaSapXep.Add(DanhSachThamSo[i]);

}

}

Mangchuasapxep();

batdaubtn.Enabled = true;

}

1. Sự kiện venutbnt\_Click: Tạo ngẫu nhiên một mảng các Node và cặp nhật lại giá trị cho Mảng chưa sắp xếp.

private void venutbtn\_Click(object sender, EventArgs e)

{

if (SoLuongNode < 2 || SoLuongNode > ThamSo.SoluongNodeLonNhat)

{

if (isEnglish == true)

{

setLang("en-US");

}

else

setLang("vi-VN");

MessageBox.Show(strSoLuongNode + ThamSo.SoluongNodeLonNhat);

soluongNodetbx.Focus();

return;

}

VeNut();

Mangchuasapxep();

}

1. Hàm DieuChinhConTrol: Khi chương trình đang được thực thi thì không cho điều chỉnh các Button trên chương trình để tạo một phiên làm việc.

private void DieuChinhControls(bool isRunning)

{

if (isRunning == true)

{

daydangxepListbox.Items.Clear();

daydangxepListbox.BringToFront();

thuattoanpanel.Enabled = khoitaopanel.Enabled = ngonngupanel.Enabled = batdaubtn.Enabled = Loaisapxeppanel.Enabled = HienThiPanel.Enabled = false;

}

else

{

thuattoanpanel.Enabled = khoitaopanel.Enabled = ngonngupanel.Enabled = batdaubtn.Enabled = Loaisapxeppanel.Enabled = HienThiPanel.Enabled = true;

}

}

1. Hàm SetMauAllControl: Để set màu của các Controls

private void setMauAllControl()

{

this.BackColor = Color.White;

batdaubtn.BackColor = dungbtn.BackColor = huybnt.BackColor = taongaunhienbtn.BackColor

= venutbtn.BackColor = docfilebtn.BackColor = nhaptaybtn.BackColor = xoamangbtn.BackColor = thoatbtn.BackColor = Color.White;

DebugPanel.BackColor = thuattoanpanel.BackColor = Loaisapxeppanel.BackColor = panel10.BackColor = HienThiPanel.BackColor

= dieukhienpanel.BackColor = khoitaopanel.BackColor = huypanel.BackColor = ngonngupanel.BackColor = Color.White;

codeListBox.BackColor = chuasapxepPanel.BackColor =yTuongTextBox.BackColor= Color.White;

}

* Hàm Mangchuasapxep: Cập nhật giá trị của mảng được tạo để sắp xếp.

private void Mangchuasapxep()

{

chuasapxepPanel.Controls.Clear();

lbMangChuaSapXep.Location = new Point(chuasapxepPanel.Location.X,10);

chuasapxepPanel.Controls.Add(lbMangChuaSapXep);

string \_string = "";

foreach (var item in DanhSachThamSo)

{

\_string += item + " ";

}

Label lbtext = new Label() { Text = \_string };

lbtext.Size = new Size(600, 20);

lbtext.Location = new Point(lbMangChuaSapXep.Location.X+lbMangChuaSapXep.Width+10, 12);

chuasapxepPanel.Controls.Add(lbtext);

}

### **3. Các hàm bắt đầu, hủy quá trình sắp xếp và các hàm hỗ trợ**

|  |
| --- |
|  |

* Hàm để bắt đầu quá trình sắp xếp:
  1. Hàm KhoiTaoMacDinh : khởi tạo giá trị mặc định khi mở chương trình.

|  |
| --- |
| private void KhoiTaoMacDinh()  {  interchangerdbtn.Checked = true;  tangrdbtn.Checked = true;  isRunning = false;  soluongNodetbx.Text = "5";  SoLuongNode = 5;  VeNut();  DieuChinhControls(isRunning);  setMauAllControl();  } |

* + Sự kiện batdaubtn\_Click: Cho phép chương trình bắt đầu thực hiện một phiên làm việc.

private void batdaubtn\_Click(object sender, EventArgs e)

{

daydangxepListbox.Visible = true;

huybnt.Enabled = true;

isRunning = true;

DieuChinhControls(isRunning);

Reset\_CountTime();

timer1.Start();

ChonThuatToan();

sapxepThread = new Thread(new ThreadStart(ThuatToanSapXep));

sapxepThread.Start();

}

* + Hàm ChonThuatToan : hàm thực hiện chọn thuật toán sắp xếp để bắt đầu một phiên làm việc.

private void ChonThuatToan()

{

if (interchangerdbtn.Checked == true)

{

ThuatToanSapXep = InterchangeSort;

}

if (bubblerdbtn.Checked == true)

{

ThuatToanSapXep = ThuatToanSapXep = BubbleSort;

}

if (selectionrdbtn.Checked == true)

{

ThuatToanSapXep = SelectionSort;

}

if (insertionrdbtn.Checked == true)

{

ThuatToanSapXep = InsertionSort;

}

if (shakerrdbtn.Checked == true)

{

ThuatToanSapXep = ShakerSort;

}

if (shellrdbtn.Checked == true)

{

ThuatToanSapXep = ShellSort;

}

if (heaprdbtn.Checked == true)

{

ThuatToanSapXep = HeapSort;

}

if (quickrdbtn.Checked == true)

{

ThuatToanSapXep = QuickSort;

}

if (binaryinsertionrdbtn.Checked == true)

{

ThuatToanSapXep = BinaryInsertionSort;

}

if (mergerdbtn.Checked == true)

{

ThuatToanSapXep = MergeSort;

}

}

* Sự kiện huybnt\_Click: hủy bỏ quá trình đang sắp xếp và tạo lại một mảng Node mới, cập nhật giá trị mảng chưa sắp xếp, Reset\_CountTime.

private void huybnt\_Click(object sender, EventArgs e)

{

sapxepThread.Abort();

VeNut();

isRunning = false;

for (int i = 0; i < SoLuongNode; i++)

{

DanhSachNode[i].Text = MangChuaSapXep[i].ToString();

DanhSachThamSo[i] = MangChuaSapXep[i];

}

DieuChinhControls(isRunning);

Mangchuasapxep();

Reset\_CountTime();

timer1.Stop();

huybnt.Enabled = false;

}

* Hàm Reset\_CountTime: Reset lại bộ đếm giờ của chương trình.

private void Reset\_CountTime()

{

Phut = Giay = 0;

label11.Text = "00:00";

}

### **4. Các hàm di chuyển vị trí của Node và hoán vị hai Node**

* Hàm DiChuyenNodeDen : hàm này di chuyển một Node đến một vị trí chỉ định

|  |
| --- |
| public void DiChuyenNodeDen(object oNode, object oVitriMoi)  {  int vitriMoi = (int)oVitriMoi;  Node node = oNode as Node;  if (vitriMoi > node.vitriHienTai)  {  node.ChuyenLen();  node.ChuyenNgang(vitriMoi);  node.ChuyenXuong();  }  else if (vitriMoi < node.vitriHienTai)  {  node.ChuyenXuong();  node.ChuyenNgang(vitriMoi);  node.ChuyenLen();  }  // set lai vi tri moi  node.vitriHienTai = vitriMoi;  } |

* Hàm HoanVi2Node : hàm này giúp đổi chổ hai Node trong mảng. Hàm tạo ra hai Task với mỗi task di chuyển một Node đến vị trí của Node còn lại. Khi thực hiện xong nó sẽ đổi màu hai Node và đổi vị trí của hai Node bên trong nodeArr

|  |
| --- |
| Task hoanVi1Task;  Task hoanVi2Task;  private void DichuyenCacNode(int vitriNodeA, int vitriNodeB)  {  DanhSachNode[vitriNodeA].BackColor = DanhSachNode[vitriNodeB].BackColor = ThamSo.mauNodeDangSX;  // Cách dùng task  hoanVi1Task = Task.Factory.StartNew(() => { DiChuyenNodeDen(DanhSachNode[vitriNodeA], vitriNodeB); });  hoanVi2Task = Task.Factory.StartNew(() => { DiChuyenNodeDen(DanhSachNode[vitriNodeB], vitriNodeA); });  // note: Task.Factory.StartNew = THÊM task vào cuối hàng đợi và CHẠY ngay khi có thể  Task.WaitAll(hoanVi1Task, hoanVi1Task);  // Thay đổi vị trí của node trong mảng nodeArray  if (DanhSachNode.Count != 0) //check xem nếu mảng còn tồn tại --> trong trường hợp mảng đã bị hủy  {  CapNhatDanhSachNode(vitriNodeA, vitriNodeB);  }  // Đổi màu 2 node sau khi sắp xếp  DanhSachNode[vitriNodeA].BackColor = DanhSachNode[vitriNodeB].BackColor = ThamSo.mauNodeHTSX;  } |

### **5. Hàm tạm dừng quá trình sắp xếp**

* Hàm TamDung : hàm này kiểm tra xem Node có đang di chuyển hay không nếu không thì tạm dừng Node lại còn có thì cho Node tiếp tục di chuyển. Hàm này điều khiển ManualResetEvent của cả Node và ManualResetEvent có trong hàm chọn từng dòng code.

|  |
| --- |
| public static ManualResetEvent codeListBoxPauseStatus = new ManualResetEvent(true);  public static bool CodeListBoxIsPause = false;  private void TamDung()  {  if (Node.IsPause)  {  Node.pauseStatus.Set(); // hàm để resume  Node.IsPause = false;  dungbtn.Text = "Tạm dừng";  timer1.Start();  }  else  {  Node.pauseStatus.Reset(); // hàm để pause  Node.IsPause = true;  dungbtn.Text = "Tiếp tục";  timer1.Stop();  }  } |

### **6. Hàm chọn từng dòng code trong khung Code C/C++**

* Hàm ChayCodeC : hàm nhận giá trị vị trí hàng được chọn trong khung Code C/C++ và chọn hàng đó.

|  |
| --- |
| public static void ChayCodeC(int line)  {  Thread.Sleep(ThamSo.ThoiGianDoi\*50);  codeListBox.SelectedIndex = line;  } |

### **7. Cập nhật trạng thái ý tưởng thuật toán**

- Khi bắt đầu quá trình sắp xếp, tùy thuộc vào phương pháp sắp xếp mà sẽ có những biến và số lượng biến khác nhau phù hợp cho phương pháp sắp xếp đó. Trong quá trình sắp xếp, những phần tử của dãy số sẽ đổi chỗ cho nhau tùy thuộc vào thuật toán mà người sử dụng chọn. Do đó ý tưởng thuật toán sẽ cập nhật liên tục trong quá trình sắp xếp.

* Hàm addYTuong : Dựa vào ngôn ngữ của giao diện để thêm vào yTuongThuatToan

|  |
| --- |
| private static void addYTuong()  {  yTuongThuatToan.Clear();  if (Mainform.isEnglish == true)  {  yTuongThuatToan.Text = yTuongEn;  }  else if (Mainform.isEnglish == false)  {  yTuongThuatToan.Text = yTuongVi;  }  } |

+ Hàm Mangchuasapxep :Cập nhật dãy số chưa sắp xếp tương ứng với dãy số ban đầu khi mới khởi tạo.

|  |
| --- |
| private void Mangchuasapxep()  {  chuasapxepPanel.Controls.Clear();  string \_string = "";  foreach (var item in DanhSachThamSo)  {  \_string += item + " ";  }  Label lbtext = new Label() { Text = \_string };  lbtext.Size = new Size(6000, 20);  lbtext.Location = new Point(30, 12);  chuasapxepPanel.Controls.Add(lbtext);  } |

### **8. Các hàm sắp xếp mảng**

* Hàm hoanTatSapXep: Xuất ra thông báo khi hoàn tất sắp xếp, Enable các button trên giao diện chương trình bị vô hiệu hóa khi chương trình đang sắp xếp.

private void hoanTatSapXep()

{

HienThiThuatToan.ChayCodeC(1);

foreach (Node item in DanhSachNode)

{

item.BackColor = ThamSo.mauNodeHTSX;

}

isRunning = false;

DieuChinhControls(isRunning);

if (DanhSachNode.Count !=0)

{

if (isEnglish == true)

{

setLang("en-US");

}

else

setLang("vi-VN");

MessageBox.Show(hoanTat, hoanTatMessageBoxName, MessageBoxButtons.OK, MessageBoxIcon.Information);

}

timer1.Stop();

sapxepThread.Abort();

foreach (Label label in bienArr.Values)

{

label.Visible = false;

}

if (!isRunning)

{

timer1.Stop();

}

huybnt.Enabled = false;

}

* Hàm InterchangeSort : sắp xếp mảng theo thuật toán InterchangeSort theo hai hướng tăng và giảm. Có thể hiện các biến trong thuật toán. Có điều chỉnh màu của các Node

|  |
| --- |
| private void InterchangeSort()  {  int i = 0, j = 0;  bool flag;  HienThiThuatToan.ChayCodeC(2);  for (i = 0; i < SoLuongNode - 1; i++)  {  bienArr["i"].Location = new Point(danhSachLabel[i].Location.X, danhSachLabel[i].Location.Y - 20);  bienArr["i"].Text = "i = " + i;  sapxepPanel.Controls.Add(bienArr["i"]);  bienArr["i"].Visible = true;  HienThiThuatToan.ChayCodeC(3);  for (j = i + 1; j < SoLuongNode; j++)  {  bienArr["j"].Location = new Point(danhSachLabel[j].Location.X, danhSachLabel[j].Location.Y - 40);  bienArr["j"].Text = "j = " + j;  sapxepPanel.Controls.Add(bienArr["j"]);  bienArr["j"].Visible = true;  HienThiThuatToan.ChayCodeC(4);    flag = false;  if (tangrdbtn.Checked == true)  {  HienThiThuatToan.ChayCodeC(5);  if (DanhSachThamSo[j] < DanhSachThamSo[i]) flag = true;  }  if (giamrdbtn.Checked == true)  {  HienThiThuatToan.ChayCodeC(5);  if (DanhSachThamSo[j] > DanhSachThamSo[i]) flag = true;  }  if (flag)  {  HienThiThuatToan.ChayCodeC(6);  Thread.Sleep(ThamSo.ThoiGianDoi);  CapNhatThamSo(i, j);  DichuyenCacNode(j, i);  Hienthimangdangsapxep(i, j, "i", "j");  }  }  }  DanhSachNode[0].BackColor = ThamSo.mauNodeHTSX;  DanhSachNode[DanhSachNode.Count - 1].BackColor = ThamSo.mauNodeHTSX;  hoanTatSapXep();  } |

* Phần code sẽ được cập nhật vào khung Code C/C++:

|  |
| --- |
| public static void InterchangeSort(bool tang = true)  {  //Vietnamese  yTuongVi =  @"Xuất phát từ đầu dãy, tìm tất các các nghịch thế chứa phần tử này, triệt tiêu chúng bằng cách đổi chỗ 2 phần tử trong cặp nghịch thế. Lặp lại xử lý trên với phần tử kế trong dãy.";  //English  yTuongEn = @"• Find the smallest number in the set and interchange it with the number in the first position.  • Then find the second smallest number, and swap it with the number in the second position.  • So on.";  //Thêm yTuong vào yTuongThuatToan  addYTuong();  string[] code = ChuyenText(  @"void InterchangeSort( int a[], int N)  {  int i, j;  for(i = 0; i < N - 1; i++)  for(j = i + 1; j < N; j++)  if( a[j] < a[i] )  Swap( a[i], a[j]);  }  ");  //thêm code vào codeListBox  codeListBox.Items.Clear();  foreach (string item in code)  {  codeListBox.Items.Add(item);  }  //nếu sắp giảm thì sửa lại  if (!tang)  {  codeListBox.Items[5] = " if( a[j] > a[i] )";  }  } |

* Hàm SelectionSort:

|  |
| --- |
| private void SelectionSort()  {  int min, i = 0, j = 0;  bool flag = false;  HienThiThuatToan.ChayCodeC(2);  for (i = 0; i < SoLuongNode - 1; i++)  {  HienThiThuatToan.ChayCodeC(3);  HienThiThuatToan.ChayCodeC(5);  bienArr["i"].Location = new Point(danhSachLabel[i].Location.X, danhSachLabel[i].Location.Y - 20);  bienArr["i"].Text = "i = " + i;  sapxepPanel.Controls.Add(bienArr["i"]);  bienArr["i"].Visible = true;  for (j = i + 1; j < SoLuongNode; j++)  {  HienThiThuatToan.ChayCodeC(6);  flag = false;  min = i;  bienArr["min"].Location = new Point(danhSachLabel[min].Location.X, danhSachLabel[min].Location.Y - 140);  bienArr["min"].Text = "min = " + min;  bienArr["min"].SendToBack();  sapxepPanel.Controls.Add(bienArr["min"]);  bienArr["min"].Visible = true;  if (tangrdbtn.Checked == true)  {  HienThiThuatToan.ChayCodeC(7);  bienArr["j"].Location = new Point(danhSachLabel[j].Location.X, danhSachLabel[j].Location.Y - 40);  bienArr["j"].Text = "j = " + j;  sapxepPanel.Controls.Add(bienArr["j"]);  bienArr["j"].Visible = true;  if (DanhSachThamSo[j] < DanhSachThamSo[min])  {  min = j;  flag = true;  bienArr["min"].Location = new Point(danhSachLabel[min].Location.X, danhSachLabel[min].Location.Y - 140);  bienArr["min"].Text = "min = " + min;  bienArr["min"].SendToBack();  }  }  if (giamrdbtn.Checked == true)  {  HienThiThuatToan.ChayCodeC(7);  bienArr["j"].Location = new Point(danhSachLabel[j].Location.X, danhSachLabel[j].Location.Y - 40);  bienArr["j"].Text = "j = " + j;  sapxepPanel.Controls.Add(bienArr["j"]);  if (DanhSachThamSo[j] > DanhSachThamSo[min])  {  min = j;  flag = true;  bienArr["min"].Location = new Point(danhSachLabel[min].Location.X, danhSachLabel[min].Location.Y - 140);  bienArr["min"].Text = "min = " + min;  bienArr["min"].SendToBack();  }  }  if (flag == true)  {  HienThiThuatToan.ChayCodeC(8);  HienThiThuatToan.ChayCodeC(9);  Thread.Sleep(ThamSo.ThoiGianDoi);  DanhSachNode[i].BackColor = DanhSachNode[min].BackColor = ThamSo.mauNodeDangSX;  CapNhatThamSo(min, i);  DichuyenCacNode(min, i);  Hienthimangdangsapxep(i, j, "i", "j");  }  }  }  DanhSachNode[0].BackColor = ThamSo.mauNodeHTSX;  DanhSachNode[DanhSachNode.Count - 1].BackColor = ThamSo.mauNodeHTSX;  hoanTatSapXep();  } |

* Phần code sẽ được cập nhật vào khung Code C/C++:

public static void SelectionSort(bool tang = true)

{

//vietnamese

yTuongVi =

@"Chọn phần tử nhỏ nhất trong N phần tử trong dãy hiện hành ban đầu.

Đưa phần tử này về vị trí đầu dãy hiện hành.

Xem dãy hiện hành chỉ còn N-1 phần tử của dãy hiện hành ban đầu.

Bắt đầu từ vị trí thứ 2;

Lặp lại quá trình trên cho dãy hiện hành... đến khi dãy hiện hành chỉ còn 1 phần tử.";

//English

yTuongEn = @"• find the smallest element

• put it in the first position

• find the next smallest element in the remaining elements

• put it in the second position

• …

• And so on, until we get to the end of the array";

//Thêm yTuong vào yTuongThuatToan

addYTuong();

string[] code = ChuyenText(

@"void SelecttionSort(int arr[], int N)

{

int min, i, j;

for (i=0; i < N-1; i++)

{

min = i;

for (j=i+1; j <N; j++)

if (a[j] < a[min])

min=j;

Swap(a[min], a[i]);

}

}");

//thêm code vào codeListBox

codeListBox.Items.Clear();

foreach (string item in code)

{

codeListBox.Items.Add(item);

}

//Nếu sắp xếp giảm dần thì sửa lại

if (!tang)

{

codeListBox.Items[7] = " if (a[j] > a[min])";

}

}

* Hàm InsertionSort

|  |
| --- |
| private void InsertionSort()  {  int pos, key;  Node Nodetam;  DanhSachNode[0].BackColor = ThamSo.mauNodeDangSX;  for (int i = 1; i < SoLuongNode; i++)  {  HienThiThuatToan.ChayCodeC(5);  DanhSachNode[i].BackColor = ThamSo.mauNodeDangSX;  key = DanhSachThamSo[i];  HienThiThuatToan.ChayCodeC(7);  Nodetam = DanhSachNode[i];  pos = i - 1;  bienArr["i"].Location = new Point(danhSachLabel[i].Location.X, danhSachLabel[i].Location.Y - 20);  bienArr["i"].Text = "i = " + i;  sapxepPanel.Controls.Add(bienArr["i"]);  bienArr["i"].Visible = true;  DanhSachNode[i].ChuyenLen();  DanhSachNode[0].BackColor = ThamSo.mauNodeHTSX;  HienThiThuatToan.ChayCodeC(8);  bienArr["pos"].Location = new Point(danhSachLabel[pos].Location.X, danhSachLabel[pos].Location.Y - 200);  bienArr["pos"].Text = "pos = " + pos;  sapxepPanel.Controls.Add(bienArr["pos"]);  bienArr["pos"].Visible = true;  if (tangrdbtn.Checked == true)  {  while ((pos >= 0) && (DanhSachThamSo[pos] > key))  {  DanhSachNode[pos].ChuyenNgang(pos + 1);  DanhSachNode[pos].vitriHienTai = pos + 1;  DanhSachNode[pos + 1] = DanhSachNode[pos];  DanhSachThamSo[pos + 1] = DanhSachThamSo[pos];  Hienthimangdangsapxep(pos, pos + 1, "pos", "i");  HienThiThuatToan.ChayCodeC(10);  pos--;  }  }  else  {  while ((pos >= 0) && (DanhSachThamSo[pos] < key))  {  DanhSachNode[pos].ChuyenNgang(pos + 1);  DanhSachNode[pos].vitriHienTai = pos + 1;  DanhSachNode[pos + 1] = DanhSachNode[pos];  DanhSachThamSo[pos + 1] = DanhSachThamSo[pos];  Hienthimangdangsapxep(pos, pos + 1, "pos", "i");  HienThiThuatToan.ChayCodeC(10);  pos--;  }  }  Nodetam.ChuyenNgang(pos + 1);  Nodetam.ChuyenXuong();  DanhSachThamSo[pos + 1] = key;  DanhSachNode[pos + 1] = Nodetam;  Nodetam.vitriHienTai = pos + 1;  DanhSachNode[pos + 1].BackColor = ThamSo.mauNodeHTSX;  }  DanhSachNode[0].BackColor = ThamSo.mauNodeHTSX;  DanhSachNode[DanhSachNode.Count - 1].BackColor = ThamSo.mauNodeHTSX;  hoanTatSapXep();  } |

* Phần code sẽ được cập nhật vào khung Code C/C++:

public static void InsertionSort(bool tang = true)

{

//Vietnamese

yTuongVi =

@"Giả sử có một dãy a0, a1,... ,an-1 trong đó i phần tử đầu tiên a0, a1,... ,ai-1 đã có thứ tự.

Tìm cách chèn phần tử ai vào vị trí thích hợp của đoạn đã được sắp để có dãy mới a0, a1,... ,ai trở nên có thứ tự. Vị trí này chính là vị trí giữa hai phần tử ak-1 và ak thỏa ak-1 < ai < ak(1≤k≤i).";

//English

yTuongEn = @"• Start by considering the first two elements of the array data. If found out of order, swap them

• Consider the third element; insert it into the proper position among the first three elements.

• Consider the fourth element; insert it into the proper position among the first four elements.

• … …";

// Thêm yTuong vào yTuongThuatToan

addYTuong();

string[] code = ChuyenText(

@"void InsertionSort(int a[], int N)

{

int pos, i;

int x;

for(i = 1; i < N; i++)

{

x = a[i]; pos = i - 1;

while((pos >= 0) && (x < a[pos]))

{

a[pos + 1] = a[pos];

pos--;

}

a[pos + 1] = x;

}

}");

// Thêm code vào codeListBox

codeListBox.Items.Clear();

foreach (string item in code)

{

codeListBox.Items.Add(item);

}

if (!tang)

{

codeListBox.Items[7] = " while((pos >= 0) && (x > a[pos]))";

}

}

* Hàm BinaryInsertionSort :

|  |
| --- |
| public void BinaryInsertionSort()  {  yTuongTextBox.Clear();  int left, right, m, i, pos;  int x;  Node nodeTam, nodeTam2;  for (i = 1; i < DanhSachNode.Count; i++)  {  pos = i - 1;  HienThiThuatToan.ChayCodeC(4);  DanhSachNode[0].BackColor = ThamSo.mauNodeHTSX;  bienArr["i"].Location = new Point(danhSachLabel[i].Location.X, danhSachLabel[i].Location.Y - 20);  bienArr["i"].Text = "i = " + i;  sapxepPanel.Controls.Add(bienArr["i"]);  bienArr["i"].Visible = true;  nodeTam = DanhSachNode[i];  x = DanhSachThamSo[i];  HienThiThuatToan.ChayCodeC(6);  DanhSachNode[i].ChuyenLen();  DanhSachNode[i].BackColor = ThamSo.mauNodeDangSX;  left = 0;  right = i - 1;  bienArr["left"].Location = new Point(danhSachLabel[left].Location.X, danhSachLabel[left].Location.Y - 140);  bienArr["left"].Text = "left = " + left;  sapxepPanel.Controls.Add(bienArr["left"]);  bienArr["pos"].Location = new Point(danhSachLabel[pos].Location.X, danhSachLabel[pos].Location.Y - 180);  bienArr["pos"].Text = "pos = " + pos;  sapxepPanel.Controls.Add(bienArr["pos"]);  bienArr["right"].Location = new Point(danhSachLabel[right].Location.X, danhSachLabel[right].Location.Y - 160);  bienArr["right"].Text = "right = " + right;  sapxepPanel.Controls.Add(bienArr["right"]);  HienThiThuatToan.ChayCodeC(7);  while (left <= right)  {  m = (left + right) / 2;  if (tangrdbtn.Checked == true)  {  if (x < DanhSachThamSo[m])  {  HienThiThuatToan.ChayCodeC(10);  bienArr["m"].Location = new Point(danhSachLabel[i].Location.X, danhSachLabel[i].Location.Y - 200);  bienArr["m"].Text = "m = " + i;  sapxepPanel.Controls.Add(bienArr["m"]);  bienArr["right"].Text = "right = " + right;  sapxepPanel.Controls.Add(bienArr["right"]);  right = m - 1;  bienArr["right"].Text = "right = " + right;  sapxepPanel.Controls.Add(bienArr["right"]);  }  else  {  HienThiThuatToan.ChayCodeC(11);  bienArr["m"].Text = "m = " + i;  sapxepPanel.Controls.Add(bienArr["m"]);  left = m + 1;  }  }  else  {  if (x > DanhSachThamSo[m])  {  HienThiThuatToan.ChayCodeC(10);  bienArr["m"].Text = "m = " + i;  sapxepPanel.Controls.Add(bienArr["m"]);  bienArr["right"].Text = "right = " + right;  sapxepPanel.Controls.Add(bienArr["right"]);  right = m - 1;  bienArr["right"].Text = "right = " + right;  sapxepPanel.Controls.Add(bienArr["right"]);  }  else  {  HienThiThuatToan.ChayCodeC(11);  bienArr["m"].Text = "m = " + i;  sapxepPanel.Controls.Add(bienArr["m"]);  left = m + 1;  }  }  HienThiThuatToan.ChayCodeC(7);  bienArr["left"].Text = "left = " + left;  sapxepPanel.Controls.Add(bienArr["left"]);  }  HienThiThuatToan.ChayCodeC(13);  bienArr["right"].Text = "right = " + right;  sapxepPanel.Controls.Add(bienArr["right"]);  pos = i - 1;  bienArr["pos"].Text = "pos = " + pos;  sapxepPanel.Controls.Add(bienArr["pos"]);  for (pos = i - 1; pos >= left; pos--)  {  bienArr["pos"].Text = "pos = " + pos;  sapxepPanel.Controls.Add(bienArr["pos"]);  bienArr["pos"].Visible = true;  HienThiThuatToan.ChayCodeC(14);  DanhSachNode[pos].ChuyenNgang(pos + 1);  DanhSachNode[pos].vitriHienTai = pos + 1;  nodeTam2 = DanhSachNode[pos + 1];  DanhSachNode[pos + 1] = DanhSachNode[pos];  DanhSachThamSo[pos + 1] = DanhSachThamSo[pos];  DanhSachNode[pos] = nodeTam2;  }  bienArr["pos"].Text = "pos = " + pos;  sapxepPanel.Controls.Add(bienArr["pos"]);  bienArr["pos"].Visible = true;  HienThiThuatToan.ChayCodeC(15);  nodeTam.ChuyenNgang(pos + 1);  nodeTam.ChuyenXuong();  DanhSachNode[pos + 1] = nodeTam;  nodeTam.vitriHienTai = pos + 1;  DanhSachNode[pos+1].BackColor = ThamSo.mauNodeHTSX;  DanhSachThamSo[left] = x;  //Cập nhật ý tưởng Text Box  Hienthimangdangsapxep(pos, i, "pos", "i");  }  DanhSachNode[0].BackColor = ThamSo.mauNodeHTSX;  DanhSachNode[DanhSachNode.Count - 1].BackColor = ThamSo.mauNodeHTSX;  hoanTatSapXep();  } |

* Phần code sẽ được cập nhật vào khung Code C/C++:

public static void BinaryInsertionSort(bool tang = true)

{

//Vietnamese

yTuongVi =

@"Cải tiến của giải thuật InsertionSort. Giải thuật chèn nhị phân cho phép giảm số lần so sánh, nhưng không làm giảm số lần dời chỗ.";

//English

yTuongEn = @"• Use binary search to reduce the number of comparisons in normal insertion sort.

• Binary Insertion Sort find use binary search to find the proper location to insert the selected item at each iteration.";

//Thêm yTuong vào yTuongThuatToan

addYTuong();

string[] code = ChuyenText(

@"void BinaryInsertionSort(int a[], int N)

{

int left, right, m, i , pos;

int x;

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

{

x = a[i]; left = 0; right = i - 1;

while(left <= right)

{

m = (left + right)/2;

if(x < a[m]) right = m - 1;

else left = m + 1;

}

for(pos = i - 1; pos >= left; pos--)

a[pos+1] = a[pos];

a[left] = x;

}

}");

//thêm code vào codeListBox

codeListBox.Items.Clear();

foreach (string item in code)

{

codeListBox.Items.Add(item);

}

//nếu sắp giảm thì sửa lại

if (!tang)

{

codeListBox.Items[11] = " if(x > a[m]) right = m - 1;";

}

}

* Hàm BubbleSort :

|  |
| --- |
| private void BubbleSort()  {  int i, j;  bool flag = false;  HienThiThuatToan.ChayCodeC(2);  HienThiThuatToan.ChayCodeC(3);  for (i = 0; i < SoLuongNode - 1; i++)  {  HienThiThuatToan.ChayCodeC(4);  bienArr["i"].Location = new Point(danhSachLabel[i].Location.X, danhSachLabel[i].Location.Y - 20);  bienArr["i"].Text = "i = " + i;  sapxepPanel.Controls.Add(bienArr["i"]);  bienArr["i"].Visible = true;  for (j = SoLuongNode - 1; j > i; j--)  {  flag = false;  if (tangrdbtn.Checked == true)  {  HienThiThuatToan.ChayCodeC(5);  if (DanhSachThamSo[j] < DanhSachThamSo[j - 1]) flag = true;  bienArr["j"].Location = new Point(danhSachLabel[j].Location.X, danhSachLabel[j].Location.Y - 40);  bienArr["j"].Text = "j = " + j;  sapxepPanel.Controls.Add(bienArr["j"]);  bienArr["j"].Visible = true;  }  if (giamrdbtn.Checked == true)  {  HienThiThuatToan.ChayCodeC(5);  if (DanhSachThamSo[j] > DanhSachThamSo[j - 1]) flag = true;  bienArr["j"].Location = new Point(danhSachLabel[j].Location.X, danhSachLabel[j].Location.Y - 40);  bienArr["j"].Text = "j = " + j;  sapxepPanel.Controls.Add(bienArr["j"]);  bienArr["j"].Visible = true;  }  if (flag == true)  {  HienThiThuatToan.ChayCodeC(6);  Thread.Sleep(ThamSo.ThoiGianDoi);  DanhSachNode[j].BackColor = DanhSachNode[j - 1].BackColor = Color.Green;  CapNhatThamSo(j, j - 1);  DichuyenCacNode(j, j - 1);  Hienthimangdangsapxep(i, j, "i", "j");  }  }  }  DanhSachNode[0].BackColor = ThamSo.mauNodeHTSX;  DanhSachNode[DanhSachNode.Count - 1].BackColor = ThamSo.mauNodeHTSX;  hoanTatSapXep();  } |

* Phần code sẽ được cập nhật vào khung Code C/C++:

public static void BubbleSort(bool tang = true)

{

//Vietnamese

yTuongVi =

@"Xuất phát từ cuối dãy, đổi chỗ các cặp phần tử kế cận để đưa phần tử nhỏ hơn trong cặp phần tử đó về vị trí đúng đầu dãy hiện hành, sau đó sẽ không xét đến nó ở bước tiếp theo, do vậy ở lần xử lý thứ i sẽ có vị trí đầu dãy là i.

Lặp lại xử lý trên cho đến khi không còn cặp phần tử nào để xét.";

//English

yTuongEn = @"• Exchange neighboring items until the largest item reaches the end of the array.

• Repeat the above step for the rest of the array.";

//Thêm yTuong vào yTuongThuatToan

addYTuong();

string[] code = ChuyenText(

@"void BubbleSort(int a[], int N)

{

int i,j;

for(i = 0; i < N - 1; i++)

for(j = N - 1; j > i; j--)

if(a[j] < a[j - 1])

Swap(a[j], a[j - 1]);

}

");

//thêm code vào codeListBox

codeListBox.Items.Clear();

foreach (string item in code)

{

codeListBox.Items.Add(item);

}

//nếu sắp giảm thì sửa lại

if (!tang)

{

codeListBox.Items[5] = " if(a[j] > a[j - 1])";

}

}

* Hàm ShakerSort :

|  |
| --- |
| private void ShakerSort()  {  HienThiThuatToan.ChayCodeC(2);  int j, left, right, k;  HienThiThuatToan.ChayCodeC(3);  left = 0; right = DanhSachThamSo.Count - 1; k = DanhSachThamSo.Count - 1;  HienThiThuatToan.ChayCodeC(4);  bienArr["left"].Location = new Point(danhSachLabel[left].Location.X, danhSachLabel[left].Location.Y - 140);  bienArr["left"].Text = "left = " + left;  sapxepPanel.Controls.Add(bienArr["left"]);  bienArr["left"].Visible = true;  bienArr["right"].Location = new Point(danhSachLabel[right].Location.X, danhSachLabel[right].Location.Y - 160);  bienArr["right"].Text = "right = " + right;  sapxepPanel.Controls.Add(bienArr["right"]);  bienArr["right"].Visible = true;  bienArr["k"].Location = new Point(danhSachLabel[k].Location.X, danhSachLabel[k].Location.Y - 60);  bienArr["k"].Text = "k = " + k;  sapxepPanel.Controls.Add(bienArr["k"]);  bienArr["k"].Visible = true;  while (left < right)  {  HienThiThuatToan.ChayCodeC(6);  j = right;  bienArr["j"].Location = new Point(danhSachLabel[j].Location.X, danhSachLabel[j].Location.Y - 40);  bienArr["j"].Text = "j = " + j;  sapxepPanel.Controls.Add(bienArr["j"]);  bienArr["j"].Visible = true;  bienArr["j"].SendToBack();  for (j = right; j > left; j--)  {  bool thucHien = false;  if (tangrdbtn.Checked == true)  {  if (DanhSachThamSo[j] < DanhSachThamSo[j - 1])  {  thucHien = true;  }  }  else  {  if (DanhSachThamSo[j] > DanhSachThamSo[j - 1])  {  thucHien = true;  }  }  HienThiThuatToan.ChayCodeC(7);  if (thucHien)  {  HienThiThuatToan.ChayCodeC(9);  CapNhatThamSo(j, j - 1);  DichuyenCacNode(j, j - 1);  Hienthimangdangsapxep(j - 1, j, "k", "j"); // mang dang sap xep  HienThiThuatToan.ChayCodeC(10);  k = j;  bienArr["k"].Location = new Point(danhSachLabel[k].Location.X, danhSachLabel[k].Location.Y - 60);  bienArr["k"].Text = "k = " + k;  sapxepPanel.Controls.Add(bienArr["k"]);  }  HienThiThuatToan.ChayCodeC(6);  bienArr["k"].Location = new Point(danhSachLabel[k].Location.X, danhSachLabel[k].Location.Y - 60);  bienArr["k"].Text = "k = " + k;  sapxepPanel.Controls.Add(bienArr["k"]);  }  HienThiThuatToan.ChayCodeC(12);  left = k;  bienArr["left"].Location = new Point(danhSachLabel[left].Location.X, danhSachLabel[left].Location.Y - 140);  bienArr["left"].Text = "left = " + left;  sapxepPanel.Controls.Add(bienArr["left"]);  HienThiThuatToan.ChayCodeC(13);  j = left;  bienArr["j"].Location = new Point(danhSachLabel[j].Location.X, danhSachLabel[j].Location.Y - 40);  bienArr["j"].Text = "j = " + j;  sapxepPanel.Controls.Add(bienArr["j"]);  for (j = left; j < right; j++)  {  bool thucHien = false;  if (tangrdbtn.Checked == true)  {  if (DanhSachThamSo[j + 1] < DanhSachThamSo[j])  thucHien = true;  }  else  {  if (DanhSachThamSo[j + 1] > DanhSachThamSo[j])  thucHien = true;  }  HienThiThuatToan.ChayCodeC(14);  if (thucHien)  {  HienThiThuatToan.ChayCodeC(16);  CapNhatThamSo(j, j + 1);  DichuyenCacNode(j + 1, j);  Hienthimangdangsapxep(j, j + 1, "k", "j");  HienThiThuatToan.ChayCodeC(17);  k = j;  bienArr["k"].Location = new Point(danhSachLabel[k].Location.X, danhSachLabel[k].Location.Y - 60);  bienArr["k"].Text = "k = " + k;  sapxepPanel.Controls.Add(bienArr["k"]);  }  HienThiThuatToan.ChayCodeC(13);  bienArr["j"].Location = new Point(danhSachLabel[j].Location.X, danhSachLabel[j].Location.Y - 40);  bienArr["j"].Text = "j = " + j;  sapxepPanel.Controls.Add(bienArr["j"]);  }  HienThiThuatToan.ChayCodeC(19);  right = k;  bienArr["right"].Location = new Point(danhSachLabel[right].Location.X, danhSachLabel[right].Location.Y - 160);  bienArr["right"].Text = "right = " + right;  sapxepPanel.Controls.Add(bienArr["right"]);  HienThiThuatToan.ChayCodeC(4);  bienArr["left"].Location = new Point(danhSachLabel[left].Location.X, danhSachLabel[left].Location.Y - 140);  bienArr["left"].Text = "left = " + left;  sapxepPanel.Controls.Add(bienArr["left"]);  bienArr["right"].Location = new Point(danhSachLabel[right].Location.X, danhSachLabel[right].Location.Y - 160);  bienArr["right"].Text = "right = " + right;  sapxepPanel.Controls.Add(bienArr["right"]);  }  HienThiThuatToan.ChayCodeC(4);  DanhSachNode[0].BackColor = ThamSo.mauNodeHTSX;  DanhSachNode[DanhSachNode.Count - 1].BackColor = ThamSo.mauNodeHTSX;  hoanTatSapXep();  } |

* Phần code sẽ được cập nhật vào khung Code C/C++:

public static void ShakerSort(bool tang = true)

{

//Vietnamese

yTuongVi = @"Trong mỗi lần sắp xếp, duyệt mảng theo 2 lượt từ 2 phía khác nhau:

Lượt đi: đẩy phần tử nhỏ về đầu mảng.

Lượt về: đẩy phần tử lớn về cuối mảng.

Ghi nhận lại những đoạn đã sắp xếp nhằm tiết kiệm các phép so sánh thừa.";

//English

yTuongEn = @"A variant of the bubble sort method, in shaker sort, n elements are sorted in n/2 phases:

• Each phase of shaker sort consists of a left to right bubbling pass followed by a right to left bubbling pass.

• In a bubbling pass pairs of adjacent elements are compared and swapped if they are out of order.";

//thêm ytuong

addYTuong();

string[] code = ChuyenText(

@"void ShakerSort(int a[], int N)

{

int j, left, right, k;

left = 0; right = N - 1, k = N - 1;

while(left < right)

{

for(j = right; j > left; j--)

if(a[j] < a[j - 1])

{

Swap(a[j], a[j - 1]);

k = j;

}

left = k;

for(j = left; j < right; j++)

if(a[j + 1] < a[j])

{

Swap(a[j], a[j + 1]);

k = j;

}

right = k;

}

}");

//thêm code vào codeListBox

codeListBox.Items.Clear();

foreach (string item in code)

{

codeListBox.Items.Add(item);

}

// nếu sắp giảm thì sửa lại

if (!tang)

{

codeListBox.Items[7] = " if(a[j] > a[j - 1])";

codeListBox.Items[14] = " if(a[j + 1] > a[j])";

}

}

* Hàm HeapSort :

|  |
| --- |
| private void HeapSort()  {  HienThiThuatToan.ChayCodeC(1);  HienThiThuatToan.ChayCodeC(2);  CreateHeap(DanhSachThamSo.Count);  HienThiThuatToan.ChayCodeC(3);  HienThiThuatToan.ChayCodeC(4);  int r = DanhSachThamSo.Count - 1;  HienThiThuatToan.ChayCodeC(5);  while (r > 0)  {  HienThiThuatToan.ChayCodeC(6);  HienThiThuatToan.ChayCodeC(7);  CapNhatThamSo(0, r);  DichuyenCacNode(r, 0);  HienThiThuatToan.ChayCodeC(8);  bienArr["right"].Location = new Point(danhSachLabel[r].Location.X, danhSachLabel[r].Location.Y - 160);  bienArr["right"].Text = "right = " + r;  sapxepPanel.Controls.Add(bienArr["right"]);  bienArr["right"].Visible = true;  bienArr["right"].SendToBack();  r--;  HienThiThuatToan.ChayCodeC(9);  if (r > 0)  {  HienThiThuatToan.ChayCodeC(10);  Shift(0, r);  }  HienThiThuatToan.ChayCodeC(11);  HienThiThuatToan.ChayCodeC(5);  }  HienThiThuatToan.ChayCodeC(12);  DanhSachNode[0].BackColor = ThamSo.mauNodeHTSX;  DanhSachNode[DanhSachNode.Count - 1].BackColor = ThamSo.mauNodeHTSX;  if (DanhSachNode.Count != 0)  {  hoanTatSapXep();  }  }  private void CreateHeap(int N)  {  HienThiThuatToan.ChayCodeC(14);  HienThiThuatToan.ChayCodeC(15);  HienThiThuatToan.ChayCodeC(16);  HienThiThuatToan.ChayCodeC(17);  int l = N / 2 - 1;  HienThiThuatToan.ChayCodeC(18);  while (l >= 0)  {  HienThiThuatToan.ChayCodeC(19);  HienThiThuatToan.ChayCodeC(20);  Shift(l, N - 1);  HienThiThuatToan.ChayCodeC(21);  bienArr["left"].Location = new Point(danhSachLabel[l].Location.X, danhSachLabel[l].Location.Y - 140);  bienArr["left"].Text = "left = " + l;  sapxepPanel.Controls.Add(bienArr["left"]);  bienArr["left"].Visible = true;  bienArr["left"].SendToBack();  l--;  HienThiThuatToan.ChayCodeC(22);  HienThiThuatToan.ChayCodeC(18);  }  HienThiThuatToan.ChayCodeC(23);  }  private void Shift(int l, int r)  {  HienThiThuatToan.ChayCodeC(25);  HienThiThuatToan.ChayCodeC(26);  HienThiThuatToan.ChayCodeC(27);  HienThiThuatToan.ChayCodeC(28);  int i = l;  int j = 2 \* i + 1;  HienThiThuatToan.ChayCodeC(29);  while (j <= r)  {  HienThiThuatToan.ChayCodeC(30);  HienThiThuatToan.ChayCodeC(31);  if (tangrdbtn.Checked == true)  {  if (j < r && DanhSachThamSo[j] < DanhSachThamSo[j + 1])  {  HienThiThuatToan.ChayCodeC(32);  j++;  }  }  else  {  if (j < r && DanhSachThamSo[j] > DanhSachThamSo[j + 1])  {  HienThiThuatToan.ChayCodeC(32);  j++;  }  }  HienThiThuatToan.ChayCodeC(33);  if (tangrdbtn.Checked == true)  {  if (DanhSachThamSo[i] < DanhSachThamSo[j])  {  HienThiThuatToan.ChayCodeC(34);  HienThiThuatToan.ChayCodeC(35);  DichuyenCacNode(j, i);  CapNhatThamSo(i, j);  Hienthimangdangsapxep(i, j, "i", "j");  HienThiThuatToan.ChayCodeC(36);  i = j;  HienThiThuatToan.ChayCodeC(37);  j = 2 \* i + 1;  HienThiThuatToan.ChayCodeC(38);  }  else  {  HienThiThuatToan.ChayCodeC(39);  return;  }  }  else  {  if (DanhSachThamSo[i] > DanhSachThamSo[j])  {  HienThiThuatToan.ChayCodeC(34);  HienThiThuatToan.ChayCodeC(35);  CapNhatThamSo(i, j);  DichuyenCacNode(i, j);  Hienthimangdangsapxep(i, j, "i", "j");  HienThiThuatToan.ChayCodeC(36);  i = j;  HienThiThuatToan.ChayCodeC(37);  j = 2 \* i + 1;  HienThiThuatToan.ChayCodeC(38);  }  else  {  HienThiThuatToan.ChayCodeC(39);  return;  }  }  HienThiThuatToan.ChayCodeC(40);  HienThiThuatToan.ChayCodeC(29);  }  HienThiThuatToan.ChayCodeC(41);  } |

* Phần code sẽ được cập nhật vào khung Code C/C++:

public static void HeapSort(bool tang = true)

{

//Vietnamese

yTuongVi =

@"Heap Sort tận dụng được các phép so sánh ở bước i-1, mà thuật toán sắp xếp chọn trực tiếp (Selectin Sort) không tận dụng được.

Để làm được điều này Heap sort thao tác dựa trên cây.";

//English

yTuongEn = @"• Transform the array of items into a heap.

• Invoke the “retrieve & delete” operation repeatedly, to extract the largest item remaining in the heap, until the heap is empty. Store each item retrieved from the heap into the array from back to front.";

//Thêm yTuong vào yTuongThuatToan

addYTuong();

string[] code = ChuyenText(

@"void HeapSort(int a[], int N)

{

CreateHeap(a,N - 1);

int r;

r = N - 1;

while(r >= 0)

{

Swap(a[0], a[r]);

r--;

if(r > 0 )

Shift(a,0,r);

}

}

void CreateHeap(int a[], int N)

{

int l;

l = N/2 - 1;

while(l >= 0)

{

Shift(a,l,N - 1);

l--;

}

}

void Shift(int a[], int l, int r)

{

int i = l;

int j = 2\*i + 1;

while(j <= r)

{

if(j < r && a[j] < a[j+1])

j++;

if(a[i] < a[j])

{

Swap(a[i], a[j]);

i = j;

j = 2\*i + 1;

}

else return;

}

}");

//thêm code vào codeListBox

codeListBox.Items.Clear();

foreach (string item in code)

{

codeListBox.Items.Add(item);

}

//nếu sắp giảm thì sửa lại

if (!tang)

{

}

}

* Hàm QuickSort :

|  |
| --- |
| public void QuickSort()  {  ThucHienQuickSort(0, DanhSachThamSo.Count - 1);  DanhSachNode[0].BackColor = ThamSo.mauNodeHTSX;  DanhSachNode[DanhSachNode.Count - 1].BackColor = ThamSo.mauNodeHTSX;  hoanTatSapXep();  }  private void ThucHienQuickSort(int left, int right)  {  HienThiThuatToan.ChayCodeC(1);  int i, j, x, vt\_x;  bienArr["left"].Text = "left = " + left;  bienArr["left"].Location = new Point(danhSachLabel[left].Location.X, danhSachLabel[left].Location.Y - 140);  sapxepPanel.Controls.Add(bienArr["left"]);  bienArr["left"].Visible = true;  bienArr["left"].SendToBack();  bienArr["right"].Text = "right = " + right;  bienArr["right"].Location = new Point(danhSachLabel[right].Location.X, danhSachLabel[right].Location.Y - 160);  sapxepPanel.Controls.Add(bienArr["right"]);  bienArr["right"].Visible = true;  bienArr["right"].SendToBack();  x = DanhSachThamSo[(left + right) / 2];  HienThiThuatToan.ChayCodeC(3);  vt\_x = (left + right) / 2;  i = left; j = right;  do  {  int z\_vt\_x = vt\_x;  if (tangrdbtn.Checked == true)  {  HienThiThuatToan.ChayCodeC(7);  bienArr["vt\_x"].Location = new Point(danhSachLabel[vt\_x].Location.X, danhSachLabel[vt\_x].Location.Y - 60);  bienArr["vt\_x"].Text = "x = a[" + vt\_x + "]";  sapxepPanel.Controls.Add(bienArr["vt\_x"]);  bienArr["vt\_x"].Visible = true;  bienArr["vt\_x"].SendToBack();  bienArr["i"].Text = "i = " + i;  bienArr["i"].Location = new Point(danhSachLabel[i].Location.X, danhSachLabel[i].Location.Y - 20);  sapxepPanel.Controls.Add(bienArr["i"]);  bienArr["i"].Visible = true;  bienArr["i"].SendToBack();  bienArr["j"].Text = "j = " + j;  bienArr["j"].Location = new Point(danhSachLabel[j].Location.X, danhSachLabel[j].Location.Y - 40);  sapxepPanel.Controls.Add(bienArr["j"]);  bienArr["j"].Visible = true;  bienArr["j"].SendToBack();  while (DanhSachThamSo[i] < x)  {  HienThiThuatToan.ChayCodeC(8);  int f\_i = i;  i++;  bienArr["i"].Text = "i = " + i;  bienArr["i"].Location = new Point(danhSachLabel[i].Location.X, danhSachLabel[i].Location.Y - 20);  sapxepPanel.Controls.Add(bienArr["i"]);  bienArr["i"].Visible = true;  bienArr["i"].SendToBack();  }  HienThiThuatToan.ChayCodeC(9);  while (DanhSachThamSo[j] > x)  {  HienThiThuatToan.ChayCodeC(10);  int f\_j = j;  j--;  bienArr["j"].Text = "j = " + j;  bienArr["j"].Location = new Point(danhSachLabel[j].Location.X, danhSachLabel[j].Location.Y - 40);  sapxepPanel.Controls.Add(bienArr["j"]);  bienArr["j"].Visible = true;  bienArr["j"].SendToBack();  }  }  else  {  HienThiThuatToan.ChayCodeC(7);  bienArr["vt\_x"].Location = new Point(danhSachLabel[vt\_x].Location.X, danhSachLabel[vt\_x].Location.Y - 60);  bienArr["vt\_x"].Text = "x = a[" + vt\_x + "]";  sapxepPanel.Controls.Add(bienArr["vt\_x"]);  bienArr["vt\_x"].Visible = true;  bienArr["vt\_x"].SendToBack();  bienArr["i"].Text = "i = " + i;  bienArr["i"].Location = new Point(danhSachLabel[i].Location.X, danhSachLabel[i].Location.Y - 20);  sapxepPanel.Controls.Add(bienArr["i"]);  bienArr["i"].Visible = true;  bienArr["i"].SendToBack();  bienArr["j"].Text = "j = " + j;  bienArr["j"].Location = new Point(danhSachLabel[j].Location.X, danhSachLabel[j].Location.Y - 40);  sapxepPanel.Controls.Add(bienArr["j"]);  bienArr["j"].Visible = true;  bienArr["j"].SendToBack();  while (DanhSachThamSo[i] > x)  {  HienThiThuatToan.ChayCodeC(8);  int f\_i = i;  i++;  bienArr["i"].Text = "i = " + i;  bienArr["i"].Location = new Point(danhSachLabel[i].Location.X, danhSachLabel[i].Location.Y - 20);  sapxepPanel.Controls.Add(bienArr["i"]);  bienArr["i"].Visible = true;  bienArr["i"].SendToBack();  }  HienThiThuatToan.ChayCodeC(9);  while (DanhSachThamSo[j] < x)  {  HienThiThuatToan.ChayCodeC(10);  int f\_j = j;  j--;  bienArr["j"].Text = "j = " + j;  bienArr["j"].Location = new Point(danhSachLabel[j].Location.X, danhSachLabel[j].Location.Y - 40);  sapxepPanel.Controls.Add(bienArr["j"]);  bienArr["j"].Visible = true;  bienArr["j"].SendToBack();  }  }  HienThiThuatToan.ChayCodeC(11);  if (i <= j)  {  int f\_vt\_x = vt\_x;  if (i == vt\_x)  {  vt\_x = j;  }  else if (j == vt\_x)  {  vt\_x = i;  }  HienThiThuatToan.ChayCodeC(13);  CapNhatThamSo(i, j);  DichuyenCacNode(j, i);  Hienthimangdangsapxep(i, j, "i", "j");  bienArr["vt\_x"].Location = new Point(danhSachLabel[vt\_x].Location.X, danhSachLabel[vt\_x].Location.Y - 60);  bienArr["vt\_x"].Text = "x = a[" + vt\_x + "]";  sapxepPanel.Controls.Add(bienArr["vt\_x"]);  bienArr["vt\_x"].Visible = true;  bienArr["vt\_x"].SendToBack();  i++; j--;  }  HienThiThuatToan.ChayCodeC(16);  } while (i <= j);  HienThiThuatToan.ChayCodeC(17);  if (left < j)  {  HienThiThuatToan.ChayCodeC(18);  ThucHienQuickSort(left, j);  }  HienThiThuatToan.ChayCodeC(19);  if (i < right)  {  HienThiThuatToan.ChayCodeC(20);  ThucHienQuickSort(i, right);  }  } |

* Phần code sẽ được cập nhật vào khung Code C/C++:

public static void QuickSort(bool tang = true)

{

//Vietnamese

yTuongVi =

@"Giải thuật QuickSort sắp xếp dãy a1, a2 ..., aN dựa trên việc phân hoạch dãy ban đầu thành 3 phần :

Phần 1:Gồm các phần tử có giá trị bé hơn x

Phần 2: Gồm các phần tử có giá trị bằng x

Phần 3: Gồm các phần tử có giá trị lớn hơn x

Với x là giá trị của một phần tử tùy ý trong dãy ban đầu.";

//English

yTuongEn = @"If the number of elements in S is 0 or 1, then

return (base case).

• Pick any element v in S (called the pivot).

• Partition the elements in S except v into two

disjoint groups:

• S1 = {x ∈ S – {v} | x ≤ v}

• S2 = {x ∈ S – {v} | x ≥ v}

• Return {QuickSort(S1

) + v + QuickSort(S2

)}";

//Thêm yTuong vào yTuongThuatToan

addYTuong();

string[] code = ChuyenText(

@"void QuickSort(int a[], int left, int right)

{

int i, j, x;

x = a[(left + night)/2];

i = left, j = right;

do

{

while(a[i] < x)

i++;

while(x < a[j])

j--;

if(i <= j)

{

Swap(a[i], a[j]);

i++, j--;

}

}while(i <= j);

if(left < j)

QuickSort(a, left, j);

if(i < right)

QuickSort(a, i, right);

}");

//thêm code vào codeListBox

codeListBox.Items.Clear();

foreach (string item in code)

{

codeListBox.Items.Add(item);

}

//nếu sắp giảm thì sửa lại

if (!tang)

{

codeListBox.Items[7] = " while(a[i] > x)";

codeListBox.Items[9] = " while(x > a[j])";

}

}

* Hàm MergeSort :

|  |
| --- |
| List<Node> b = new List<Node>();  List<Node> c = new List<Node>();  int nb, nc;  int Min(int a, int b)  {  if (a > b) return b;  else return a;  }  void Distribute(List<Node> a, int N, ref int nb, ref int nc, int k)  {  int i, pa, pb, pc;  pa = pb = pc = 0;  HienThiThuatToan.ChayCodeC(10);  while (pa < N)  {  HienThiThuatToan.ChayCodeC(12);  for (i = 0; (pa < N) && (i < k); i++, pa++, pb++)  {  HienThiThuatToan.ChayCodeC(13);  a[pa].BackColor = ThamSo.mauNodeDangSX;  a[pa].ChuyenLen();  a[pa].ChuyenNgang(pb);  a[pa].vitriHienTai = pb;  b[pb] = a[pa];  HienThiThuatToan.ChayCodeC(12);  }  HienThiThuatToan.ChayCodeC(12);  HienThiThuatToan.ChayCodeC(14);  for (i = 0; (pa < N) && (i < k); i++, pa++, pc++)  {  HienThiThuatToan.ChayCodeC(15);  a[pa].BackColor = ThamSo.mauNodeDangSX;  a[pa].ChuyenXuong();  a[pa].ChuyenNgang(pc);  a[pa].vitriHienTai = pc;  c[pc] = a[pa];  HienThiThuatToan.ChayCodeC(14);  }  HienThiThuatToan.ChayCodeC(14);  HienThiThuatToan.ChayCodeC(10);  }  HienThiThuatToan.ChayCodeC(10);  nb = pb; nc = pc;  }  void Merge(List<Node> a, int nb, int nc, int k)  {  int p, pb, pc, ib, ic, kb, kc;  p = pb = pc = 0; ib = ic = 0;  HienThiThuatToan.ChayCodeC(23);  while ((nb > 0) && (nc > 0))  {  kb = Min(k, nb);  kc = Min(k, nc);  HienThiThuatToan.ChayCodeC(26);  bool thucHien = false; // dùng để xét tăng/giảm , nếu bằng true thì code sẽ chạy  if (tangrdbtn.Checked == true)  {  if (c[pc + ic].Value >= b[pb + ib].Value)  thucHien = true;  }  else  {  if (c[pc + ic].Value <= b[pb + ib].Value)  thucHien = true;  }  if (thucHien)  {  HienThiThuatToan.ChayCodeC(28);  b[pb + ib].BackColor = ThamSo.mauNen;  b[pb + ib].ChuyenXuong();  b[pb + ib].ChuyenNgang(p);  b[pb + ib].vitriHienTai = p;  a[p] = b[pb + ib];  p = p + 1;  ib++;  HienThiThuatToan.ChayCodeC(29);  if (ib == kb)  {  HienThiThuatToan.ChayCodeC(31);  for (; ic < kc; ic++)  {  HienThiThuatToan.ChayCodeC(32);  c[pc + ic].BackColor = ThamSo.mauNen;  c[pc + ic].ChuyenLen();  c[pc + ic].ChuyenNgang(p);  c[pc + ic].vitriHienTai = p;  a[p] = c[pc + ic];  p = p + 1;  HienThiThuatToan.ChayCodeC(32);  }  pb += kb; pc += kc; ib = ic = 0;  nb -= kb; nc -= kc;  }  }  else  {  HienThiThuatToan.ChayCodeC(39);  c[pc + ic].BackColor = ThamSo.mauNen;  c[pc + ic].ChuyenLen();  c[pc + ic].ChuyenNgang(p);  c[pc + ic].vitriHienTai = p;  a[p] = c[pc + ic];  p = p + 1;  ic++;  HienThiThuatToan.ChayCodeC(40);  if (ic == kc)  {  HienThiThuatToan.ChayCodeC(42);  for (; ib < kb; ib++)  {  HienThiThuatToan.ChayCodeC(43);  b[pb + ib].BackColor = ThamSo.mauNen;  b[pb + ib].ChuyenXuong();  b[pb + ib].ChuyenNgang(p);  b[pb + ib].vitriHienTai = p;  a[p] = b[pb + ib];  p = p + 1;  HienThiThuatToan.ChayCodeC(43);  }  HienThiThuatToan.ChayCodeC(42);  pb += kb; pc += kc; ib = ic = 0;  nb -= kb; nc -= kc;  }  }  HienThiThuatToan.ChayCodeC(23);  }  HienThiThuatToan.ChayCodeC(23);  if (a.Count % 2 == 1 && (k != (a.Count - 1)))  {  if (nb > nc)  {  HienThiThuatToan.ChayCodeC(43);  b[pb].BackColor = ThamSo.mauNen;  b[pb].ChuyenXuong();  b[pb].ChuyenNgang(a.Count - 1);  b[pb].vitriHienTai = a.Count - 1;  }  }  if (a.Count % 2 == 0 && Math.Abs(nb - nc) == 2)  {  HienThiThuatToan.ChayCodeC(43);  b[pb].BackColor = ThamSo.mauNen;  b[pb].ChuyenXuong();  b[pb].ChuyenNgang(a.Count - 2);  b[pb].vitriHienTai = a.Count - 2;  HienThiThuatToan.ChayCodeC(43);  b[pb + 1].BackColor = ThamSo.mauNen;  b[pb + 1].ChuyenXuong();  b[pb + 1].ChuyenNgang(a.Count - 1);  b[pb + 1].vitriHienTai = a.Count - 1;  };  }  void ThucHienMergeSort(List<Node> a, int N)  {  for (int i = 0; i < SoLuongNode; i++)  {  b.Add(new Node(i));  c.Add(new Node(i));  }  int k;  HienThiThuatToan.ChayCodeC(53);  for (k = 1; k < N; k \*= 2)  {  bienArr["k"].Location = new Point(danhSachLabel[k].Location.X, danhSachLabel[k].Location.Y - 200);  bienArr["k"].Text = "k = " + k;  sapxepPanel.Controls.Add(bienArr["k"]);  bienArr["k"].Visible = true;  HienThiThuatToan.ChayCodeC(55);  Distribute(a, N, ref nb, ref nc, k);  HienThiThuatToan.ChayCodeC(56);  Merge(a, nb, nc, k);  HienThiThuatToan.ChayCodeC(53);  }  HienThiThuatToan.ChayCodeC(53);  }  void MergeSort()  {  yTuongTextBox.Clear();  bienArr["a:"].Location = new Point(0, 110 + DanhSachNode[0].Height / 2);  bienArr["a:"].Text = "a:";  sapxepPanel.Controls.Add(bienArr["a:"]);  bienArr["c:"].Location = new Point(0, 160 + DanhSachNode[0].Height / 2);  bienArr["c:"].Text = "c:";  sapxepPanel.Controls.Add(bienArr["c:"]);  bienArr["b:"].Location = new Point(0, 60 + DanhSachNode[0].Height / 2);  bienArr["b:"].Text = "b:";  sapxepPanel.Controls.Add(bienArr["b:"]);  bienArr["a:"].Visible = bienArr["b:"].Visible = bienArr["c:"].Visible = true;  HienThiThuatToan.ChayCodeC(51);  ThucHienMergeSort(DanhSachNode, DanhSachNode.Count);  foreach (Node node in DanhSachNode)  {  node.BackColor = ThamSo.mauNodeHTSX;  }  DanhSachNode[0].BackColor = ThamSo.mauNodeHTSX;  DanhSachNode[DanhSachNode.Count - 1].BackColor = ThamSo.mauNodeHTSX;  hoanTatSapXep();  } |

* Phần code sẽ được cập nhật vào khung Code C/C++:

public static void MergeSort(bool tang = true)

{

//Vietnamese

yTuongVi =

@"Sắp xếp dãy a(1),a(2),...,a(n) dựa trên nhận xét sau :

-Mỗi dãy a(1),a(2),...,a(n) bất kỳ là một tập hợp các dãy con liên tiếp mà mỗi dãy con đều đã có thứ tự.

-Dãy đã có thứ tự coi như có 1 dãy con.

Hướng tiếp cận : tìm cách làm giảm số dãy con không giảm của dãy ban đầu.";

//English

yTuongEn = @"• Divide the array into two halves.

• Conquer each half by sorting it recursively

• Combine the results obtained, by merging the two halves to obtain one fully sorted array";

//Thêm yTuong vào yTuongThuatToan

addYTuong();

string[] code = ChuyenText(

@"int b[MAX], c[MAX], nb, nc;

int Min(int a, int b)

{

if(a > b) return b;

else return a;

}

void Distribute(int a[], int N, int &nb, int &nc, int k)

{

int i, pa, pb, pc;

pa = pb = pc = 0 ;

while(pa < N)

{

for(i = 0; (pa < N) && (i < k); i++, pa++, pb++)

b[pb] = a[pa];

for(i = 0; (pa < N) && (i < k); i++, pa++, pc++)

c[pc] = a[pa];

}

nb = pb; nc = pc;

}

void Merge(int a[], int nb, int nc, int k)

{

int p, pb, pc, ib, ic, kb, kc;

p = pb = pc = 0; ib = ic = 0;

while((nb > 0) && (nc > 0))

{

kb = Min(k, nb); kc = Min(k, nc);

if(c[pc + ic] < b[pb + ib] == false)

{

a[p++] = b[pb + ib]; ib++;

if(ib == kb)

{

for(;ic < kc; ic++)

a[p++] = c[pc + ic];

pb += kb; pc += kc; ib = ic = 0;

nb -= kb; nc -= kc;

}

}

else

{

a[p++] = c[pc + ic]; ic++;

if(ic == kc)

{

for(;ib < kb; ib++)

a[p++] = b[pb + ib];

pb += kb; pc += kc; ib = ib = 0;

nb -= kb; nc -= kc;

}

}

}

}

void MergeSort(int a[], int N)

{

int k;

for(k = 1; k < N; k\*= 2)

{

Distribute(a, N, nb, nc, k);

Merge(a, nb, nc, k);

}

}

");

//thêm code vào codeListBox

codeListBox.Items.Clear();

foreach (string item in code)

{

codeListBox.Items.Add(item);

}

//nếu sắp giảm thì sửa lại

if (!tang)

{

codeListBox.Items[26] = " if(c[pc + ic] > b[pb + ib] == false)";

}

}

* Hàm ShellSort :

|  |
| --- |
| private void ShellSort()  {  HienThiThuatToan.ChayCodeC(1);  for (int gap = SoLuongNode / 2; gap > 0; gap /= 2)  {  HienThiThuatToan.ChayCodeC(3);  bienArr["gap"].Location = new Point(danhSachLabel[gap].Location.X, danhSachLabel[gap].Location.Y - 180);  bienArr["gap"].Text = "gap = " + gap;  sapxepPanel.Controls.Add(bienArr["gap"]);  bienArr["gap"].SendToBack();  bienArr["gap"].Visible = true;  for (int i = gap; i < SoLuongNode; i++)  {  bienArr["i"].Location = new Point(danhSachLabel[i].Location.X, danhSachLabel[i].Location.Y - 20);  bienArr["i"].Text = "i = " + i;  sapxepPanel.Controls.Add(bienArr["i"]);  bienArr["i"].Visible = true;  HienThiThuatToan.ChayCodeC(5);  if (tangrdbtn.Checked == true)  {  HienThiThuatToan.ChayCodeC(7);  for (int j = i; j >= gap && DanhSachThamSo[j] < DanhSachThamSo[j - gap]; j -= gap)  {  bienArr["j"].Location = new Point(danhSachLabel[j].Location.X, danhSachLabel[j].Location.Y - 40);  bienArr["j"].Text = "j = " + j;  bienArr["j"].SendToBack();  bienArr["j"].Visible = true;  sapxepPanel.Controls.Add(bienArr["j"]);  HienThiThuatToan.ChayCodeC(9);  Thread.Sleep(ThamSo.ThoiGianDoi);  DanhSachNode[j].BackColor = DanhSachNode[j - gap].BackColor = ThamSo.mauNodeDangSX;  CapNhatThamSo(j, j - gap);  DichuyenCacNode(j, j - gap);  Hienthimangdangsapxep(i, j, "i", "j");  }  }  if (giamrdbtn.Checked == true)  {  HienThiThuatToan.ChayCodeC(7);  for (int j = i; j >= gap && DanhSachThamSo[j] > DanhSachThamSo[j - gap]; j -= gap)  {  HienThiThuatToan.ChayCodeC(9);  Thread.Sleep(ThamSo.ThoiGianDoi);  DanhSachNode[j].BackColor = DanhSachNode[j - gap].BackColor = ThamSo.mauNodeDangSX;  CapNhatThamSo(j, j - gap);  DichuyenCacNode(j, j - gap);  Hienthimangdangsapxep(i, j, "i", "j");  }  }  }  }  DanhSachNode[0].BackColor = ThamSo.mauNodeHTSX;  DanhSachNode[DanhSachNode.Count - 1].BackColor = ThamSo.mauNodeHTSX;  hoanTatSapXep();  } |

* Phần code sẽ được cập nhật vào khung Code C/C++:

public static void ShellSort(bool tang = true)

{

//Vietnamese

yTuongVi =

@"Cải tiến của phương pháp chèn trực tiếp (Insertion Sort).

Phân hoạch dãy thành các dãy con.

Sắp xếp các dãy con theo phương pháp chèn trực tiếp.

Dùng phương pháp chèn trực tiếp sắp xếp lại cả dãy.";

//English

yTuongEn = @"• Arrange the data sequence in a two-dimensional array.

• Sort the columns of the array.";

//Thêm yTuong vào yTuongThuatToan

addYTuong();

string[] code = ChuyenText(

@"void ShellSort (int a[], int N)

{

for (int gap = N / 2; gap > 0; gap /= 2)

{

for (int i = gap; i < N; i++)

{

for (int j = i; j >= gap && a[j] < a[j - gap]; j -= gap)

{

Swap(a[j], a[j - gap]);

}

}

}

}");

//thêm code vào codeListBox

codeListBox.Items.Clear();

foreach (string item in code)

{

codeListBox.Items.Add(item);

}

//nếu sắp giảm thì sửa lại

//if (!tang)

//{

//}

**}**