Bài 1

#include <stdio.h>

void print\_multiples\_of\_seven\_two\_digits() {

printf("Các số nguyên có 2 chữ số và là bội của 7 là:\n");

for (int i = 10; i < 100; i++) {

if (i % 7 == 0) {

printf("%d ", i);

}

}

printf("\n");

}

int main() {

print\_multiples\_of\_seven\_two\_digits();

return 0;

}

Bài 2

#include <stdio.h>

#include <math.h>

// Hàm kiểm tra xem một số có phải là số chính phương hay không

int isPerfectSquare(int n) {

int sqrt\_n = sqrt(n);

return (sqrt\_n \* sqrt\_n == n);

}

// Hàm đếm số lượng số chính phương nhỏ hơn n

int countPerfectSquares(int n) {

int count = 0;

for (int i = 1; i < n; ++i) {

if (isPerfectSquare(i)) {

count++;

}

}

return count;

}

int main() {

int n;

printf("Nhap vao mot so nguyen duong n: ");

scanf("%d", &n);

printf("Cac so chinh phuong nho hon %d la:\n", n);

for (int i = 1; i < n; ++i) {

if (isPerfectSquare(i)) {

printf("%d ", i);

}

}

printf("\n");

printf("So luong cac so chinh phuong nho hon %d la: %d\n", n, countPerfectSquares(n));

return 0;

}

Bài 3

from .\_anvil\_designer import Form1Template

from anvil import \*

import anvil.server

import math

# Hàm kiểm tra và chuyển đổi chuỗi đầu vào

def parse\_input(input\_text):

return [int(x) for x in input\_text.split(',') if x.strip().isdigit()]

# Hàm Insertion Sort

def insertion\_sort(arr):

for i in range(1, len(arr)):

key = arr[i]

j = i - 1

while j >= 0 and arr[j] > key:

arr[j + 1] = arr[j]

j -= 1

arr[j + 1] = key

return arr

# Hàm Selection Sort

def selection\_sort(arr):

for i in range(len(arr)):

min\_idx = i

for j in range(i + 1, len(arr)):

if arr[j] < arr[min\_idx]:

min\_idx = j

arr[min\_idx], arr[i] = arr[i], arr[min\_idx]

return arr

# Hàm Bubble Sort

def bubble\_sort(arr):

for i in range(len(arr) - 1):

for j in range(len(arr) - i - 1):

if arr[j] > arr[j + 1]:

arr[j], arr[j + 1] = arr[j + 1], arr[j]

return arr

# Hàm Merge Sort

def merge(arr, l, m, r):

L = arr[l:m + 1]

R = arr[m + 1:r + 1]

i, j, k = 0, 0, l

while i < len(L) and j < len(R):

if L[i] <= R[j]:

arr[k] = L[i]

i += 1

else:

arr[k] = R[j]

j += 1

k += 1

while i < len(L):

arr[k] = L[i]

i += 1

k += 1

while j < len(R):

arr[k] = R[j]

j += 1

k += 1

def merge\_sort(arr, l, r):

if l < r:

m = (l + r) // 2

merge\_sort(arr, l, m)

merge\_sort(arr, m + 1, r)

merge(arr, l, m, r)

return arr

# Hàm xử lý sự kiện khi nút sắp xếp được nhấp

class Form1(Form1Template):

def \_\_init\_\_(self, \*\*properties):

self.init\_components(\*\*properties)

self.sort\_options = ["Insertion Sort", "Selection Sort", "Bubble Sort", "Merge Sort"]

self.sort\_dropdown.items = self.sort\_options

def sort\_button\_click(self, \*\*event\_args):

# Lấy dữ liệu từ TextBox

input\_text = self.numbers\_text\_box.text

# Chuyển đổi thành danh sách số nguyên

arr = parse\_input(input\_text)

# Lựa chọn thuật toán sắp xếp

sort\_choice = self.sort\_dropdown.selected\_value

# Sắp xếp dựa trên lựa chọn

if sort\_choice == "Insertion Sort":

sorted\_arr = insertion\_sort(arr)

elif sort\_choice == "Selection Sort":

sorted\_arr = selection\_sort(arr)

elif sort\_choice == "Bubble Sort":

sorted\_arr = bubble\_sort(arr)

elif sort\_choice == "Merge Sort":

merge\_sort(arr, 0, len(arr) - 1)

sorted\_arr = arr

# Hiển thị kết quả

self.result\_label.text = "Sắp xếp: " + ', '.join(map(str, sorted\_arr))