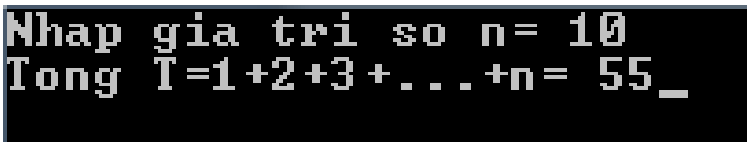
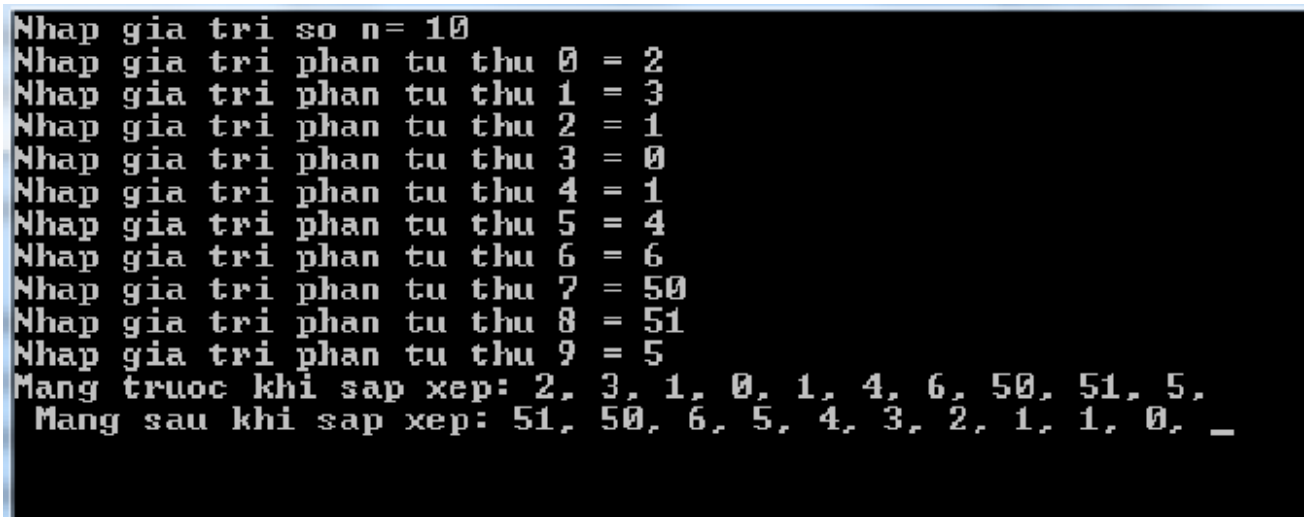
BÀI TẬP THỰC HÀNH NNLT C++

MỤC TIÊU 2: CON TRỎ VÀ MẢNG

Bài tập 1: Viết chương trình nhập vào 1 số n và tính tổng sau:
T = 1 + 2 + 3 + ... + n
Yêu cầu sử dụng hàm tính tổng sau:
Prototype: int tong (int *a, int *b): trả về tổng của 2 số mà 2 con trỏ a và b đang trỏ tới



Bài tập 2: Viết chương trình nhập vào một số n. Khởi tạo và nhập một mảng n số nguyên. Sau đó sắp xếp mảng đã cho theo thứ tự giảm dần.
Yêu cầu sử dụng 2 hàm sau:
Prototype: void swap (int *a, int *b): hoán đổi 2 giá trị của 2 con trỏ a và b đang trỏ tới.
void sap_xep(int *a, int length): sắp xếp mảng do con trỏ a trỏ tới và có chiều dài là length.



Bài tập 3: Viết chương trình khởi tạo một mảng một chiều gồm 1000 số nguyên ngẫu nhiên, sau đó thực hiện:

a. Xóa bỏ các giá trị <=0 trong mảng. Chú ý chọn cách xóa tối ưu.
b. Sắp xếp mảng theo thứ tự giảm dần. Yêu cầu sử dụng 2 hàm sau:

- Hàm `void swap (int *a, int *b)`: hoán đổi 2 giá trị mà 2 con trỏ a và b đang trỏ tới.
- Hàm `void sap_xep(int *a, int length)`: sắp xếp mảng do con trỏ a trỏ tới và có chiều dài là length.

c. Nhập một giá trị x từ bàn phím (x>0), sau đó chèn x vào mảng sao cho vẫn đảm bảo trật tự đã có.

Bài tập 4: Tìm kiếm nhị phân: Viết chương trình tìm kiếm một phần tử trong mảng đã có trật tự. Theo mô tả dưới đây:

Cho một mảng A có n phần tử với các giá trị hoặc bản ghi $A_0, A_1, A_2, \ldots, A_{n-1}$ đã được sắp xếp sao cho $A_0 \leq A_1 \leq A_2 \leq \cdots \leq A_{n-1}$, và giá trị cần tìm T, chương trình con sau đây sử dụng tìm kiếm nhị phân để tìm chỉ số của T trong A.[7]

1. Gán L với giá trị 0 và R với giá trị $n-1$.
2. Nếu $L > R$, tìm kiếm kết thúc (không thành công).
3. Gán m (vị trí của phần tử đứng giữa) với giá trị floor của $\frac{L+R}{2}$, tức là số nguyên lớn nhất nhỏ hơn hoặc bằng $\frac{L+R}{2}$.
4. Nếu $A_m < T$, gán L với $m+1$ và quay lại bước 2.
5. Nếu $A_m > T$, gán R với $m-1$ và quay lại bước 2.
6. Khi $A_m = T$, quá trình tìm kiếm hoàn tất; trả về m.

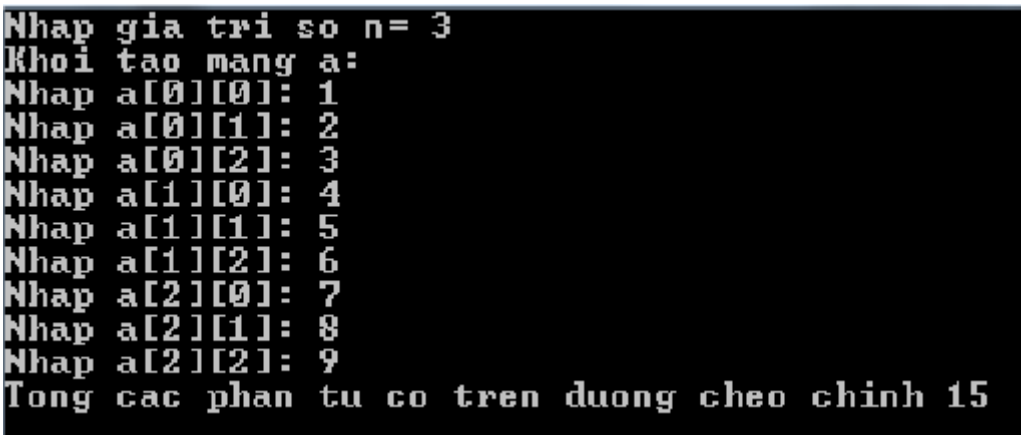
Bài tập 5: Sàng Eratosthenes – Sieve of Erathosthenes

Viết hàm in ra màn hình tất cả các số nguyên tố bé hơn 1000 và đếm số lượng các số nguyên tố ấy, dùng phương pháp Sàng Eratoshthenes với mảng 1000 số nguyên không âm đầu tiên.

(Hướng dẫn: Phương pháp Sàng Eratosthenes như sau: Biết rằng số 0 và số 1 không phải là số nguyên tố, 2 là số nguyên tố đầu tiên; từ đó về sau, để tìm các số nguyên tố ta chỉ việc loại đi các bội số của bất kỳ số nguyên tố nào đã tìm thấy trước đó, tức là:

Trước hết, loại mọi bội số của 2 (4, 6, 8, ...),
Kế đến, loại mọi bội số của 3 (6, 9, 12, ...),
Kế đến, loại mọi bội số của 5 (10, 15, 20, ..) // 4 đã được loại sẵn rồi
Và cứ thế tiếp tục.
Những số còn sót lại (không bị loại) chính là các số nguyên tố.

Bài tập 6: Viết chương trình nhập vào 1 số n. Khởi tạo và nhập vào một mảng 2 chiều gồm n hàng và n cột. Tính tổng tất cả các số có trên đường chéo chính.
Yêu cầu sử dụng 2 hàm sau:
Prototype: void nhap(int **a, int n)//nhập mảng 2 chiều [nxn] do con trỏ a trỏ tới
int tong_cheo (int **a, int n)// trả về tổng các phần tử trên đường chéo chính


Nhap gia tri so n= 3
Khoi tao mang a:
Nhap a[0][0]: 1
Nhap a[0][1]: 2
Nhap a[0][2]: 3
Nhap a[1][0]: 4
Nhap a[1][1]: 5
Nhap a[1][2]: 6
Nhap a[2][0]: 7
Nhap a[2][1]: 8
Nhap a[2][2]: 9
Tong cac phan tu co tren duong cheo chinh 15