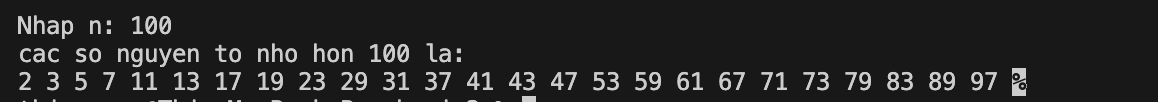
Bài tập buổi 2

**Bài 1**: Viết chương trình sử dụng hàm liệt kê các số nguyên tố, sử dụng sàn Eratosthenes trong khoảng 1,n nhập từ bàn phím.

* **Chương trình:**

|  |
| --- |
| #include <iostream>  #include <vector>  #include <math.h>  using namespace std;  void eratosthenes( int n) {  int A[n+1];  for (int i = 2; i <=n; i++) A[i]=0;  for(int i=2; i<n ; i++){  if(A[i]!=0)continue;  for (int x = 2\*i; x <=n ; x+=i)  {  A[x]=i;  }  }  cout << "cac so nguyen to nho hon "<< n << " la: " << '\n';  for (int i = 2; i <=n ; i++)  {  if (A[i]==0)  {  cout << i << " ";  }    }    }  int main() {  system("clear");  cout << "Nhap n: ";  int n;  cin >> n;  eratosthenes(n);  return 0;  } |

* **Kết quả:**



* Độ phức tạp

Khởi tạo mảng: O(n)

Sàng Eratosthenes: O(nloglogn)

In kết quả: O(n)

**Bài 2:** Viết chương trình thuật toán euclid.

* **Chương trình:**

|  |
| --- |
| #include <iostream>  #include <vector>  #include <math.h>  using namespace std;  int gcd(int a, int b){  if(b>a)gcd(b,a);  if(b==0)return a;  else return gcd(b,a%b);  }  int main(){  system("clear");  int a,b;  cout << "nhap a: ";  cin>>a;  cout << "nhap b: ";  cin >> b;  cout << " Uoc chung lon nhat cua " << a << " và "<< b << " là : ";  cout << gcd(a,b);  return 0;  } |

* **Kết quả:**

A black background with white text

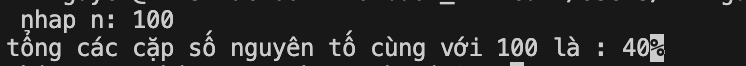
Description automatically generated

**Bài 3:** Viết chương trình  thể hiện Euler's totient function.

* **Chương trình:**

|  |
| --- |
| #include <iostream>  #include <vector>  #include <math.h>  using namespace std;  struct node {  int co\_so;  int so\_mu;  };  vector<node> phan\_ra\_snt( int n) {  int i,k = 0;  i = 2;  vector<node> v;  while (n > 1 || k>0) {  if (n % i == 0) {  k++;  n /= i;  } else {  if(k > 0 ) v.push\_back((node){i,k});  k = 0;  i++;  }  }  return v;  }  bool check\_snt( int n) {  if (n < 2) return 0;  for (int i = 2; i <= sqrt(n); i++) {  if (n % i == 0) return 0;  }  return 1;  }  int euler\_totient\_fast(int n){  int t=1;  if(check\_snt(n)) return n-1;  vector<node> arr=phan\_ra\_snt(n);  for (int i = 0; i < arr.size(); i++)  {  t \*= pow(arr[i].co\_so,arr[i].so\_mu - 1 )\*(arr[i].co\_so - 1);  }  return t;  }  int main(){  cout << " nhap n: ";  int n;  cin >> n;  cout << "tổng các cặp số nguyên tố cùng với "<< n << " là : " << euler\_totient\_fast(n);  return 0;  } |

* **Kết quả:**

****

**Bài 4:** Cho n là đủ lớn, phân tích thừa số nguyên tố với số n. Ước số lớn nhất của n là m, tìm số nguyên tố gần m nhất.

* **Chương trình:**

|  |
| --- |
| #include <iostream>  #include <vector>  #include <math.h>  using namespace std;  struct node {  int co\_so;  int so\_mu;  };  vector<node> phan\_ra\_snt( int n) {  int i,k = 0;  i = 2;  vector<node> v;  while (n > 1 || k>0) {  if (n % i == 0) {  k++;  n /= i;  } else {  if(k > 0 ) v.push\_back((node){i,k});  k = 0;  i++;  }  }  return v;  }  bool check\_snt( int n) {  if (n < 2) return 0;  for (int i = 2; i <= sqrt(n); i++) {  if (n % i == 0) return 0;  }  return 1;  }  int so\_nt\_gan\_nhat(int n){  int i=1;  do{  if( check\_snt(n-i)) return n-i;  if( check\_snt(n+i)) return n+i;  i++;  }while(1);  }  int main(){  int n;  cout << "nhap n: ";  cin >> n;  cout << "phan tich thau so nguyen to cua "<< n << " la:";  vector<node> v= phan\_ra\_snt(n);  for (int i = 0; i < v.size(); i++) {  cout << v[i].co\_so ;  if (v[i].so\_mu > 1) {  cout << "^" << v[i].so\_mu;  }  if (i < v.size() - 1) {  cout << " \* ";  }else {  cout << endl;  }  }  cout << "uoc lon nhat la: ";  int m= n/(v[0].co\_so);  cout << m << endl;  cout << " so nguyen to gan " << m << " nhat la: "<<so\_nt\_gan\_nhat(m);  return 0;  } |

* **Kết quả:**

**A black background with white text

Description automatically generated**

**Bài 5:** Tìm số hoàn hảo gần x nhất với x nhập từ bàn phím.

* **Chương trình:**

|  |
| --- |
| #include <iostream>  #include <vector>  #include <math.h>  using namespace std;  struct node {  int co\_so;  int so\_mu;  };  vector<node> phan\_ra\_snt( int n) {  int i,k = 0;  i = 2;  vector<node> v;  while (n > 1 || k>0) {  if (n % i == 0) {  k++;  n /= i;  } else {  if(k > 0 ) v.push\_back((node){i,k});  k = 0;  i++;  }  }  return v;  }  bool check\_shh(int n){  vector<node> v = phan\_ra\_snt(n);  long long result = 1;  for (int i=0; i<v.size(); i++) {  result \*= (pow(v[i].co\_so, v[i].so\_mu + 1) - 1) / (v[i].co\_so - 1);  }  cout << result;  if(result == 2\*n ) return true;  else return false;  }  int shh\_gan\_nhat(int x){  int k=1;  do  {  if (check\_shh(x+k)) return x+k;  if(check\_shh(x-k)) return x-k;  k++;    }while(1);    }  int main(){  int x;  cout << "nhap x: ";  cin >> x;  cout << "so hoan hao gan "<< x << " nhat la: " <<shh\_gan\_nhat(x);    return 0;  } |

* **Kết quả:**

**A screenshot of a phone

Description automatically generated**