**Bài 1. Tổng quan về thuật toán**

1. Thuật toán

Thuật toán là một dãy các bước biến đổi các dữ liệu đầu vào (input) thành những dữ liệu đầu ra (output) sau hữu hạn bước.

Ví dụ: ước chung lớn nhất

|  |
| --- |
| **if** \_\_name\_\_ == **'\_\_main\_\_'**:  a=int(input(**'nhap a = '**))  b=int(input(**'nhap b = '**))  **while** a!=b:  **if** a>b: a=a-b  **else**: b=b-a  print(**"uoc chung lon nhat {0:d}"**.format(a)) |

Ví dụ: ước chung lớn nhất của dãy

|  |
| --- |
| **from** functools **import** reduce **def** ucln(a,b):  **while** b:  r=a%b  a=b  b=r  **return** a **if** \_\_name\_\_ == **'\_\_main\_\_'**:  a=list(map(int,input(**"Nhap day : "**).split()))  b=reduce(ucln,a)   print(**"uoc chung lon nhat cua day"**,b) |

**Một số tính chất của thuật toán**

* ***Thuật toán chỉ xử lý thông tin thành thông tin***

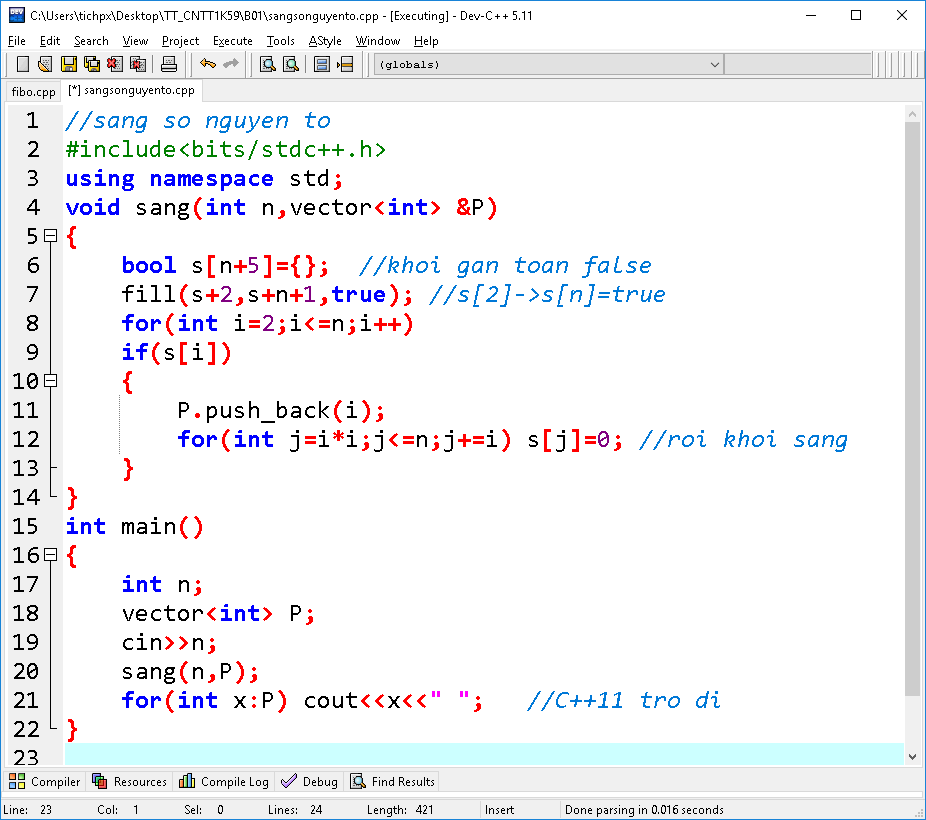
Ví dụ: Số fibonacci

|  |
| --- |
| **def** fib(n):  f=[1]\*(n+5) *#mang co n+5 so 1* **for** i **in** range (2,n+1): f[i]=f[i-1]+f[i-2]  **return** f[n] **if** \_\_name\_\_ == **'\_\_main\_\_'**:  n=int(input(**"nhap n : "**))  print(fib(n-1)) |

* ***Thuật toán có tính xác định***: sau mỗi một bước xác định duy nhất một bước tiếp theo, bước nào trước thực hiện trước, bước nào sau thực hiện sau

Ví dụ: Thuật toán sàng các số nguyên tố Eratosthens

Nhập vào n tìm các số nguyên tố nhỏ hơn hoặc bằng n



Code

|  |
| --- |
| **def** sang(n):  s=[**True**]\*(n+5)  p=[]  **for** i **in** range (2,n+1):  **if** s[i]:  *#p=p+[i]* p.append(i)  **for** j **in** range (i\*i,n+1,i): s[j]=**False  return** p  **if** \_\_name\_\_ == **'\_\_main\_\_'**:  n=int(input(**"Nhap n = "**))  p=sang(n)  *#for x in p: print(x,end=" ")* print(\*p,sep=**' '**) |

* ***Thuật toán phải có tính dừng***: Tính được kết quả sau hữu hạn bước

Ví dụ: Cho số n nó có thể sinh ra 2\*n hoặc 3\*n+1, sau đó những số này sinh tiếp đến khi gặp số nguyên tố thì không sinh nữa, hãy tìm một số nguyên tố mà tổ tiên của nó là n.



Code

|  |
| --- |
| **def** sang(n):  s=[**True**]\*(n+5)  **for** i **in** range (2,n+1):  **if** s[i]:  **for** j **in** range (i\*i,n+1,i): s[j]=**False  return** s  **if** \_\_name\_\_ == **'\_\_main\_\_'**:  n=int(input(**"Nhap n = "**))  s=sang(100000)  q=[n]  M={}  M[n]=**True  while True**:  x=q[0]  q=q[1:]  **if** 2 \* x **not in** M.keys():  M[2 \* x] = **True** q.append(2 \* x)  **if** 3\*x+1 **not in** M.keys():  M[3\*x+1]=**True** q.append(3\*x+1)  **if** s[3\*x+1]:  print(3\*x+1)  **break** |

Tóm tắt buổi 1

* Thuật toán
* Vector
* Hàm fibonacci
* Sàng số nguyên tố
* Bfs

1. Độ phức tạp thuật toán
2. Ngôn ngữ lập trình