Giải thuật/ thuật toán: tập các bước có thứ tự để giải quyết 1 vấn đề  
Đặc điểm: + các bước rõ rang = definite   
 +các bước có thứ tự=finite  
 +số các bước hữu hạn =precise and effective

+thực thi đọc lập =implementation independent

Biểu diễn : Natural language= ngôn ngữ tự nhiên

Flow chart =lưu đồ

Pseudo code=mã giả

Programming language=ngôn ngữ lập trình

Start/end

In put/out put

process

decision

Computing Ecosystem  
Computer:A computer is a machine that manipulates data according to a list of instructions.  
a computer is a device capable of performing computations and making logical decisions at speed millions and even billion of times faster them human beings can.  
Computer is an electronic machine which comes after

STEPS IN PROGRAM DEVELOPMENT

The various steps involved are:

Defining or Analyzing the problem: Xác định hoặc phân tích vấn đề

Design (Algorithm): Thiết kế (Thuật toán)

Coding: Mã hóa

Documenting the program: Lập hồ sơ chương trình

Compiling and running the program: Biên dịch và chạy chương trình

Testing and Debugging: Kiểm tra và gỡ lỗi

Maintenance: Sự bảo trì  
  
STEPS IN PROGRAM DEVELOPMENT: ANALYZING OR DEFINING THE PROBLEM  
The problem is defined by doing a preliminary investigation: Vấn đề được xác định bằng cách thực hiện một cuộc điều tra sơ bộ

Defining a problem helps us to understand problem clearly: Xác định một vấn đề giúp chúng ta hiểu vấn đề một cách rõ ràng

It is also known as Program Analysis: Nó còn được gọi là Phân tích Chương trình  
  
TASKS IN DEFINING A PROBLEM (NHIỆM VỤ XÁC ĐỊNH VẤN ĐỀ)

Followings are the tasks in order to define a problem (Tiếp theo là các nhiệm vụ để xác định một vấn đề)

Specifying the input requirements: Chỉ định các yêu cầu đầu vào

Specifying the output requirements: Chỉ định các yêu cầu đầu ra

Specifying the processing requirements:chỉ định các yêu cầu xử lý

SPECIFYING THE INPUT REQUIREMENTS (ĐẶT CỤ THỂ YÊU CẦU ĐẦU VÀO)

\* The input specification is obtained by answering following questions (Đặc điểm kỹ thuật đầu vào có được bằng cách trả lời các câu hỏi sau):

What specific values will be provided as input to the program? (Những giá trị cụ thể nào sẽ được cung cấp làm đầu vào cho chương trình?)

What format will the values be? (Định dạng các giá trị sẽ là gì?)

For each input item, what is the valid range of values that it may assume? (Đối với mỗi mục đầu vào, phạm vi giá trị hợp lệ mà nó có thể giả định là gì?)

What restrictions are placed on the use of these values? (Những hạn chế nào được đặt ra đối với việc sử dụng các giá trị này?)

\*The output specification is obtained by answering the following questions(Đặc điểm kỹ thuật đầu ra có được bằng cách trả lời như sau câu hỏi):

What values will be produced? (Những giá trị nào sẽ được sản xuất?)

What is the format of these values? (Định dạng của các giá trị này là gì?)

What specific annotation, headings, or titles are required in the report? (Chú thích, tiêu đề hoặc tiêu đề cụ thể nào được yêu cầu trong báo cáo?)

What is the amount of output that will be produced? (Sản lượng sẽ được sản xuất là bao nhiêu?)

\*The processing requirement is obtained by answering following questions(Yêu cầu xử lý đạt được bằng cách trả lời các câu hỏi sau):

What is the method (technique) required in producing the desired output?( Phương pháp (kỹ thuật) cần thiết để tạo ra sản lượng mong muốn là gì?)

What are the validation checks that need to be applied to the input data?( Kiểm tra xác thực cần được áp dụng cho dữ liệu đầu vào là gì?)

What calculations are needed?( Những tính toán nào là cần thiết?)  
  
\*STEPS IN PROGRAM DEVELOPMENT: DESIGN (CÁC BƯỚC PHÁT TRIỂN CHƯƠNG TRÌNH: THIẾT KẾ)

A design is the path from the problem to a solution in code(Thiết kế là đường dẫn từ vấn đề đến giải pháp trong mã):

The well designed program is likely to be ( Chương trình được thiết kế tốt có khả năng là):

Easier to read and understand later(Dễ dàng hơn để đọc và hiểu sau này)

Less of bugs and errors(Ít lỗi và sai sót)

Easier to extend to add new features(Dễ dàng mở rộng hơn để thêm các tính năng mới)

Easier to program in the first place(Dễ dàng lập trình hơn ngay từ đầu)

\*MODULAR DESIGN (THIẾT KẾ HIỆN ĐẠI)

Once the problem is defined clearly, several design methodologies can be applied (Một khi vấn đề được xác định rõ ràng, một số phương pháp thiết kế có thể được áp dụng):

An important approach is Top-Down program design(Một cách tiếp cận quan trọng là thiết kế chương trình Top-Down)

It is structured design technique(Đó là kỹ thuật thiết kế có cấu trúc)

It breaks up the problem into a set of sub-problems called Modules(Nó chia nhỏ vấn đề thành một tập hợp các vấn đề con được gọi là Mô-đun)

It creates a hierarchical structure of the modules(Nó tạo ra một cấu trúc phân cấp của các mô-đun)

\*STEPS IN PROGRAM DEVELOPMENT: CODING(CÁC BƯỚC PHÁT TRIỂN CHƯƠNG TRÌNH: GIẢI MÃ)

An algorithm expressed in programming languages is called Program(Một thuật toán được thể hiện bằng ngôn ngữ lập trình được gọi là Chương trình)

Writing a program is called Coding(Viết một chương trình được gọi là Coding)

The logic that has been developed in the algorithm is used to write program(Logic đã được phát triển trong thuật toán được sử dụng để viết chương trình)

\*STEPS IN PROGRAM DEVELOPMENT: DOCUMENTING THE PROGRAM(CÁC BƯỚC PHÁT TRIỂN CHƯƠNG TRÌNH: LẬP TÀI LIỆU CHƯƠNG TRÌNH)

Document explains(Tài liệu giải thích)

How the program works and how to use the program (user manual)( Cách chương trình hoạt động và cách sử dụng chương trình (hướng dẫn sử dụng))

How to maintain the program (developer manual)( Cách duy trì chương trình (hướng dẫn dành cho nhà phát triển))

Details of particular programs, or particular pieces of programs, are easily forgotten or confused without suitable documentation(Chi tiết về các chương trình cụ thể, hoặc các phần cụ thể của chương trình, là

dễ bị quên hoặc nhầm lẫn nếu không có tài liệu phù hợp)

\*FORMS OF DOCUMENTATION(Hình thức văn bản)

Documentation comes in two forms(Tài liệu có hai dạng)

External documentation, which includes things such as reference manuals,algorithm descriptions, flowcharts, and project workbooks(Tài liệu bên ngoài, bao gồm những thứ như hướng dẫn sử dụng tham chiếu, mô tả thuật toán, sơ đồ và sổ làm việc dự án)

Internal documentation, which is part of the source code itself (essentially, the declarations, statements, and comments)( Tài liệu nội bộ, là một phần của chính mã nguồn (về cơ bản, các khai báo, câu lệnh và nhận xét))

\*COMPILING AND EXECUTING THE PROGRAM

Compilation is a process of translating a source program into machine understandable form

The compiler is system software(Trình biên dịch là phần mềm hệ thống)

It examines each instruction for its correctness

It does the translation

During the execution(Trong quá trình thực hiện)

Program is loaded into the computer’s memory (Chương trình được tải vào bộ nhớ của máy tính)

The program instructions are executed (Các hướng dẫn chương trình được thực hiện)

\*Variable has():

Name: grade, age, salary(Tên: cấp, tuổi, lương)

Type: integer, float, string(Loại: số nguyên, float, string)

Data (value): 100, 10.5, “John”( Dữ liệu (giá trị): 100, 10,5, “John”)

Variable is used to store data(Biến được sử dụng để lưu trữ dữ liệu)

“Variable” means it could store different data at different time ("Biến" có nghĩa là nó có thể lưu trữ các dữ liệu khác nhau tại thời điểm khác nhau)

\*DATA TYPES(LOẠI DỮ LIỆU)

Variable must have a data type, which specifies(Biến phải có kiểu dữ liệu chỉ định)

Which kind of data it can contain(Loại dữ liệu nào nó có thể chứa)

Size of data to be stored in the memory (bytes)( Kích thước dữ liệu được lưu trữ trong bộ nhớ (byte))

How to convert from them from/to binary format(Làm thế nào để chuyển đổi chúng từ / sang định dạng nhị phân)

Data types can be divided into(Các kiểu dữ liệu có thể được chia thành)

Primitive data types e.g., int, float double, etc. (Các kiểu dữ liệu nguyên thủy, ví dụ: int, float double, v.v.)

Reference data types e.g., array, pointer, etc. (Các kiểu dữ liệu tham chiếu, ví dụ: mảng, con trỏ, v.v.)

1 số kiểu số nguyên:

Lưu các số -2^31 🡪 2^31 – 1  
chiếm 32 bit trong bộ nhớ

Float:số phẩy động  
double:số phẩy động  
string:chuỗi ký tự  
char:lưu 1 ký tự  
bool:true/false  
  
  
để hiện thị câu lệnh trên màn hình thì ta dung câu lệnh Console.Writerline console.readkey  
  
-statement:kết thúc  
-comment: +Trên 1 dòng

+ 1 khối lệnh  
**-Biến (variable):**dung để lưu dữ liệu  
1 biến có : +Tên  
 +Loại Tên biến=giá trị  
 +Dữ liệu   
-Kiểu dữ liệu (Data Types):Xđ loại dữ , XĐ kích thước DL trong bộ nhớ 1 vài kiểu DL cơ bản  
-Kiểu số nguyên   
 int (32bit) - 2^31 ⬄ 2^31 – 1  
 VD:1,2,3,4,…  
-Kiểu số phẩy động

VD:9.5 6.73  
 2.34 2.66667  
 float  
 double(độ chính xác cao hơn)  
-Kiểu chuỗi ký tự  
 String giá trị kiểu string đặt trong   
 VD: “Ha Gia Ninh” “Nguyen Ngoc Son”  
-Kiểu ký tự   
 char ‘a’ ‘b’ ‘c’  
-Kiểu logic   
 bool true or false  
  
**Toán tử**-Toán tử số học: + , - , \* , /   
% : chia lấy dư VD: 5%2=1  
 7%3=1  
++: Tăng 1 đơn vị  
--: Giảm 1 đơn vị  
-Toán tử gán   
=  
+=  
-= x+=2 ⬄ x=x+2  
\*= x-=3 ⬄ x=x-3

/=

%=  
  
^= x^=3 ⬄x=x^3  
>>= x>>=3 ⬄ dịch x về trước 3 đơn vị  
-Toán tử so sánh  
== : so sánh bằng nhau  
  
>,<,>=,<=   
  
-Toán tử logic  
  
&& ⬄ and  
ll ⬄ OR  
l ⬄ NOT

Cấu trúc điều kiện   
if   
if ……..else  
if ……else if……else  
  
Cú pháp  
 if (Điều kiện 1)

{Câu lệnh nếu đúng}

Else if(đk2) ->ngược vs ddk1 và bổ sung đk2  
 {}

Else

{Câu lệnh nếu điều kiện sai}

Toán tử điều kiện   
  
(Biểu thức điều kiện ? giá trị nếu biểu thức đúng : giá trị nếu biểu thức sai)  
  
-> Toán tử bậc ba  
  
VD:  
str=(a>=b ? “a là số lớn” : “b là số lớn”)

If …else : lồng nhau  
if(điều kiện)

{

……if( ){…..}

}  
else{…..}  
  
Câu lệnh switch(biểu thức)

{

Case giá trị 1:khối lệnh 1  
 break;

Case giá trị 2:khối lệnh 2

Break;

Case giá trị n: khối lệnh n

Break;

Depault: khối lệnh mặc định

}

Vòng lặp

-Lặp đi lặp lại 1 biểu thức

Các loại:

-for

-for each  
-while

-do ……..while

Cú pháp

For(initialization: condition ; increment/decrement)

{

//one or more statements;

}

Initialization:Khai báo và gán giá trị cho biến điều khiển vòng lặp

Condition:điều kiện lặp. Các câu lệnh sẽ lặp lại khi condition=true

Increment/decrement:toán tử tăng/giảm

1: initialization

2: Kiểm tra Condition

3:if Condition = true ->Thực hiện satements

False->dừng

4:tang/giảm biến điều khiển

vòng lặp lồng nhau

for()

{

For()

{

}

}

Vòng lặp While

While(dk)

{//code}

Do{//code}

While{dk};

Hàm (function)

Phương thức (Method)

Cú pháp khai báo

Kiểu trả về tên hàm( kiểu Dk1 tham số 1, kiểu dk2 tham số 2)

{

//code

Returm biểu thức

}