

1. MỞ ĐẦU

1.1 Lí do chọn đề tài.

Cỗ nhân đã có câu “Hiền tài là nguyên khí của Quốc Gia”. Do vậy, cùng với “nâng cao dân trí” thì “bồi dưỡng nhân tài” được Bộ Giáo Dục xác định là nhiệm vụ trọng tâm, được triển khai đồng bộ ở các cấp học phổ thông trong cả nước. Theo nội dung chương 1 – Mục tiêu và nhiệm vụ của môn Tin học thì nhiệm vụ của bộ môn Tin học đã được Bộ Giáo Dục xác định rõ: “Bộ môn Tin học phải đảm bảo chất lượng phổ cập, đồng thời phải có nhiệm vụ phát hiện và bồi dưỡng HSG Tin học, cung cấp cho đất nước những nhân tài trong lĩnh vực công nghệ thông tin”.^[1]

Trong các kì thi HSG môn Tin học của tỉnh Thanh Hóa nói riêng và các kì thi HSG Tin học nói chung, bài thi HSG môn Tin học thường được chấm bằng chương trình chấm tự động dựa trên các bộ Test chấm.

Do đó để nâng cao chất lượng bồi dưỡng HSG, trong quá trình dạy học tôi thường hướng dẫn cho học sinh làm bài tập và kiểm tra bằng các bộ Test, qua đó giúp học sinh rèn luyện kỹ năng làm bài, tự hoàn thiện dần năng lực bản thân đồng thời giúp tạo hứng thú học tập cho học sinh.

Thực tế, giáo viên rất dễ trong việc tìm kiếm các bài tập, các đề thi có kèm theo đáp án từ trên mạng Internet, nhưng rất khó để tìm các bộ Test chấm. Trong khi đó, để tự xây dựng các bộ Test chấm cho từng bài tập là công việc rất khó khăn, tốn nhiều thời gian và công sức. Do đó, tôi đã nghiên cứu, vận dụng *Tiện ích sinh test tự động* vào việc xây dựng bộ Test chấm và mang lại hiệu quả nhất định, từ đó nâng cao chất lượng bồi dưỡng HSG môn Tin học trong nhà trường.

Từ những lí do trên, tôi mạnh dạn trình bày SKKN “**Sử dụng có hiệu quả Tiện ích Sinh Test tự động vào việc xây dựng Bộ Test chấm nhằm nâng cao chất lượng bồi dưỡng HSG môn Tin học**”.

1.2 Mục đích nghiên cứu

Đề tài “**Sử dụng có hiệu quả Tiện ích sinh Test vào việc xây dựng các bộ Test chấm nhằm nâng cao chất lượng bồi dưỡng học sinh giỏi môn Tin học**”, tôi hướng tới mục đích:

Tìm ra cách vận dụng linh hoạt *Tiện ích sinh Test tự động* vào việc xây dựng Test chấm trong các bài tập lập trình, giúp công việc Tạo test chấm nhanh chóng hơn, hiệu quả hơn, từ đó phục vụ cho công tác bồi dưỡng và đánh giá chất lượng đội tuyển HSG môn Tin học.

1.3 Đối tượng nghiên cứu

Phần mềm chấm thi tự động Themis, tiện ích hỗ trợ xây dựng test chấm của Thạc sĩ Nguyễn Tô Sơn. Ngôn ngữ lập trình được tôi sử dụng trong quá trình nghiên cứu, ứng dụng là ngôn ngữ Pascal.

1.4 Phương pháp nghiên cứu

Trong quá trình nghiên cứu và hoàn thiện SKKN này, tôi đã sử dụng phối kết hợp nhiều phương pháp:

- Phương pháp nghiên cứu tài liệu: Tôi đã tiến hành nghiên cứu nhiều tài liệu như hướng dẫn sử dụng phần mềm Themis, hướng dẫn sử dụng *Tiện ích sinh test tự động*. Ngoài ra tôi còn tham khảo thêm các đề thi, đáp án, các bộ Test chấm đề thi học sinh giỏi được tôi sưu tầm từ nhiều nguồn khác nhau
- Phương pháp thực nghiệm: Trên cơ sở phát huy những mặt đã làm được, rút kinh nghiệm những mặt còn hạn chế, ở các lần tiến hành sau tôi thường có những cải tiến nhất định, giúp công việc xây dựng Test chấm hiệu quả hơn.
- Phương pháp tổng kết: Từ quá trình nghiên cứu, thực nghiệm và rút kinh nghiệm, tôi đã tổng kết những kinh nghiệm của bản thân trong việc xây dựng test chấm sao cho nhanh, chính xác và đem lại hiệu quả cao hơn.

2. NỘI DUNG

2.1 Cơ sở lí luận

Theo nội dung chương 1 – Mục tiêu và nhiệm vụ của môn Tin học thì nhiệm vụ của bộ môn Tin học đã được Bộ Giáo Dục xác định rõ: “Bộ môn Tin học phải đảm bảo chất lượng phổ cập, đồng thời phải có nhiệm vụ phát hiện và bồi dưỡng HSG Tin học, cung cấp cho đất nước những nhân tài trong lĩnh vực công nghệ thông tin”^[1]

Theo công văn Số: 2268/TB-SGDDT về cấu trúc đề thi HSG cấp tỉnh các môn văn hóa ở cấp THPT và THCS ban hành ngày 19 tháng 9 năm 2018 thì: Bài thi học sinh giỏi môn Tin học được chấm bằng chương trình chấm tự động (chạy các test), có so sánh thời gian chạy chương trình của các thí sinh để đánh giá. Chỉ xem xét văn bản chương trình để cho điểm trong các trường hợp đặc biệt. Số test của mỗi câu có thể bằng số điểm hoặc gấp đôi, ba,... số điểm và các test phải dần hướng tới tính hoàn thiện bài toán, dữ liệu lớn dần, độ phức tạp tăng dần,... ^[2]

2.2 Thực trạng

Xuất phát từ thực tế, kì thi HSG môn Tin học thường được chấm bằng máy tính, sử dụng chương trình chấm tự động dựa trên các Test. Trong quá trình tham gia bồi dưỡng HSG môn Tin học tôi nhận thấy: việc giáo viên sử dụng các bộ Test để chấm bài làm môn Tin học cho học sinh khi giải các bài tập lập trình hoặc trong các bài kiểm tra đánh giá chất lượng HSG là việc làm rất cần thiết. Việc làm này không chỉ giúp giáo viên đánh giá khả năng làm bài của học sinh, mà còn giúp học sinh biết cách phân tích đề, hoàn thiện năng lực làm bài, tạo hứng thú học tập cho học sinh.

Trước đây, để tạo được các bộ Test chấm tôi thường phải làm theo cách thủ công, công việc này thường rất vất vả, tốn nhiều thời gian và công sức. Do vậy, tôi chỉ thực hiện chấm bài làm cho học sinh trong các bài khảo sát kiểm tra chất lượng HSG, việc làm này không thường xuyên vì bản thân tôi rất ngại xây dựng Test chấm. Còn trong quá trình học, tôi chỉ hướng dẫn học sinh giải quyết các bài tập, chạy được chương trình. Sau đó kiểm tra chương trình bằng bộ Test có sẵn trong đề bài và một vài bộ test có giá trị Input nhỏ, dễ nhìn thấy kết quả Output. Khi chương trình bài làm của học sinh có kết quả Output giống với Output tôi xây dựng tôi liền vội vàng kết luận bài làm của học sinh đã đúng. Kết quả đánh giá bài làm của học sinh có thể chưa phản ánh đúng sự chính xác của thuật toán. Do đó học sinh thường bị mất điểm trong các trường hợp dữ liệu có giá trị đặc biệt và dữ liệu có giá trị lớn.

Trong 2 lần kiểm tra khảo sát chất lượng HSG lần 1 và lần 2 được tôi tiến hành đầu năm 2018 -2019, kết quả bài làm của học sinh như sau:

Kì Thi khảo sát	Họ tên	Điểm câu 1	Điểm câu 2	Điểm câu 3	Điểm câu 4	Điểm câu 5	Tổng điểm
Lần 1	Lê Thị Giang	1.5/6	2.5/5	1.5/4	0.5/3	0/2	6/20
	Vũ Văn Cường	2/6	2/5	3/4	2/3	1/2	10/20
Lần 2	Lê Thị Giang	2/6	3/4	2/4	0/3	0.5/2	7.5/20
	Vũ Văn Cường	3/6	1/5	2/4	2/3	0/2	8/20

Kết quả trên rất thấp, mặc dù ở thời điểm này cả hai đề khảo sát thời tôi ra đều không khó, đề ra vào các dạng bài đã được tôi hướng dẫn trước đó. Nhưng do bài làm của học sinh chưa được tối ưu, dẫn đến mất điểm trong một vài test. Từ kết quả hai lần thi khảo sát trên, nên trong quá trình bồi dưỡng tôi đã chú ý cho học sinh kiểm tra bài làm của mình bằng các bộ Test chấm một cách thường xuyên liên tục hơn.

Để xây dựng các bộ Test chấm tôi đã trăn trở nghiên cứu và thực hiện bằng nhiều cách khác nhau. Từ đó tôi nhận ra cách vận dụng Tiện ích sinh Test tự động sẽ đem lại hiệu quả cao nhất.

2.3 Nội dung nghiên cứu của đề tài

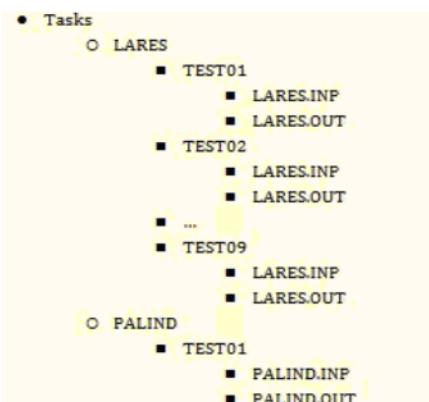
2.3.1 Thuật ngữ: Bộ Test chấm

Bộ Test: là bộ gồm giá trị Input và Output đúng của bài toán.

Bộ Test chấm được hiểu là một thư mục mẹ, bên trong có nhiều thư mục con (mỗi thư mục con là một bộ Test), bên trong thư mục con sẽ có 2 tệp: Input và output tương ứng đúng của bài toán.

Bộ Test chấm được sử dụng để đánh giá mức độ hoàn thiện bài làm của học sinh, từ đó đánh giá và cho điểm.

Trong quá trình bồi dưỡng và đánh giá HSG tại trường, để kiểm tra đánh giá bài làm của học sinh tôi thường sử dụng phần mềm chấm thi tự động Themis (*Hướng dẫn sử dụng phần mềm Themis và link download đã được tôi lưu vào đĩa CD nộp kèm theo SKKN này*). Để chấm bài bằng phần mềm Themis, bộ Test chấm yêu cầu phải được xây dựng theo đúng cấu trúc cây thư mục sau:



Hình 1: Cấu trúc cây thư mục Test chấm theo yêu cầu của phần mềm chấm tự động Themis

Vấn đề đặt ra là làm thế nào để tạo cây thư mục Test chấm như trên? Làm thế nào để tạo ra các tệp Input, Output trong mỗi thư mục con, và đảm bảo các tệp này trong các thư mục con đều có tên giống hệt nhau, chỉ khác nhau phần nội dung bên trong?

Dưới đây tôi chỉ ra 2 cách mà tôi đã từng làm, trong đó cách 2 (cách sử dụng Tiện ích) do có nhiều ưu điểm vượt trội nên hiện nay tôi chỉ sử dụng cách làm này.

2.3.2 Cách xây dựng bộ test chấm thủ công (cách làm cũ)

B1: Tự nhập dữ liệu hoặc viết chương trình để tạo ra Input của Test1.

B2: Chạy chương trình code đáp án để sinh ra output tương ứng của Test1.

B3: Di chuyển tệp Input, output trên vào cây thư mục theo cấu trúc quy định (Hình 1).

B4: Lặp lại các bước trên, cho đến khi đủ số test theo mong muốn.

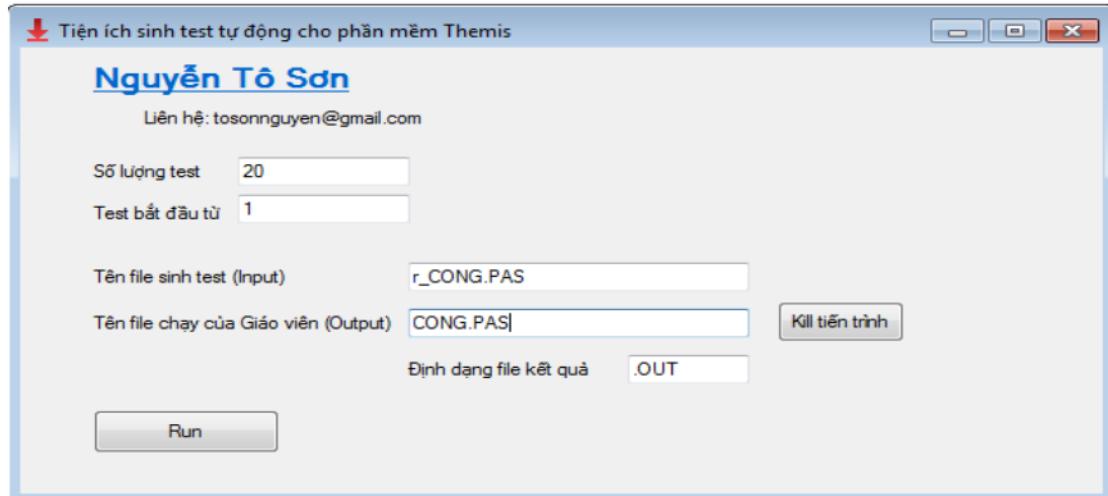
Với cách làm này, giả sử cần tạo một bộ Test chấm gồm 20 Test, thì giáo viên sẽ cần phải tạo 20 thư mục con (Test 1, test 2,). Theo đó các công việc tạo tệp Input, chạy chương trình để sinh ra output tương ứng, di chuyển các thư mục Input, Output vào đúng các thư mục Test1, test 2... cũng phải thực hiện 20 lần. Do vậy, hạn chế lớn nhất của cách làm này là giáo viên sẽ phải tốn rất nhiều thời gian, lặp đi lặp lại các công việc tương tự nhau rất nhảm chán, và dễ gây nhầm lẫn.

Từ những lí do trên, hiện nay tôi không còn tạo Test chấm theo cách làm này nữa.

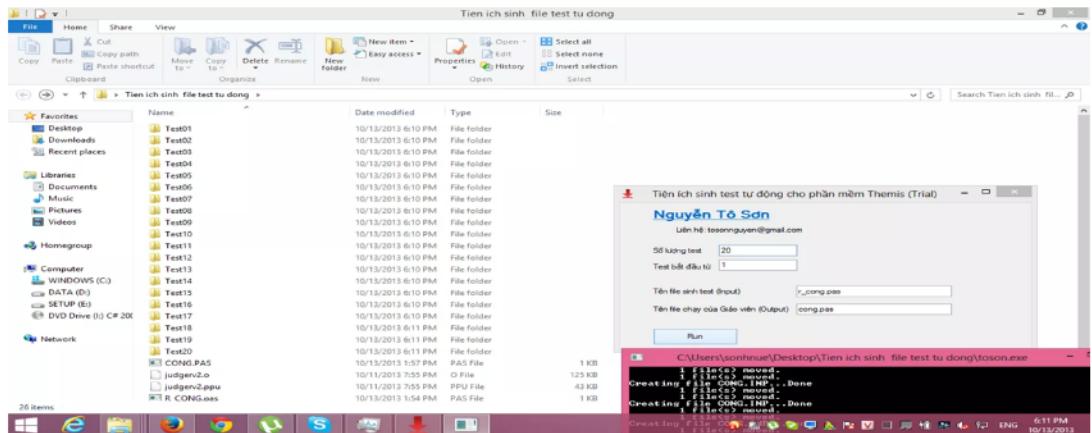
2.3.3 Cách xây dựng bộ test chấm nhờ sử dụng Tiện ích Sinh Test tự động

a) Giới thiệu tổng quan về Tiện ích sinh test tự động

Tiện ích sinh Test tự động là phần mềm được xây dựng và phát triển bởi Thầy Nguyễn Tô Sơn – Giảng Viên trường Đại Học Sư Phạm Hà Nội. Tiện ích cho phép người dùng tạo bộ Test chấm theo cây thư mục phù hợp với phần mềm chấm điểm tự động Themis – của T.S Lê Minh Hoàng.[3]



Hình 2. Giao diện Tiện ích sinh Test tự động – Thầy Nguyễn Tô Sơn



Hình 3: Hình ảnh một bộ test chấm được sinh ra từ Tiện ích Chuẩn bị về phần cứng và phần mềm

- Một máy tính đã được cài đặt Free Pascal và cài đặt biến môi trường Path (xem thêm tại [tài liệu tham khảo 4 – cách cài đặt biến môi trường](#))
- Copy file sinh test mẫu của tác giả tiện ích (file này sinh ra file .INP) và code đáp án của giáo viên (file này từ file .INP trên sinh ra file .OUT vào cùng thư mục với tiện ích ở trên. Chúng ta dùng 2 file .INP và .OUT này để chấm điểm chất lượng chương trình được viết bởi học sinh)

Lưu ý: Tiện ích này có giao diện bằng Tiếng Việt, được sử dụng miễn phí, và không cần cài đặt.

Hướng dẫn sử dụng: gồm các bước làm chính sau

Bước 1: Tạo tệp code đáp án (tệp này sẽ dùng để sinh ra file Output)

Bước 2: Tạo tệp sinh Input (tệp này sẽ dùng để sinh ra Input) bằng cách sửa lại nội dung nằm giữa 2 phần **try** và **finally** ở chương trình mẫu của tác giả (tùy thuộc vào dữ liệu của đề bài, cần sửa lại chương trình cho phù hợp).

// Chương trình Tao cac file Input cua Ban giam khao

```

{$MODE OBJFPC}
program RandomTest;
const
Prefix = 'CONG.INP';
procedure GenTest(const FName: String);
var
f: Text;
A, B, C, D: Integer;
begin
Randomize;
assign(f, FName); rewrite(f);
try
A := Random(10000) + 1;
B := Random(10000) + 1;
C := Random(10000) + 1;
D := Random(10000) + 1;
WriteLn(f, A, ' ', B);
WriteLn(f, C, ' ', D);
finally
close(f);
end;
end;
procedure Gen09_AZ;
var
c: Char;
begin
for c := '1' to '1' do
GenTest(Prefix);
end;
begin
Gen09_AZ;
end.

```

Bước 3: Chạy chương trình tiện ích.

Lúc này máy tính sẽ tự động tạo ra cây thư mục Test chấm theo cấu trúc phù hợp với cây thư mục mà phần mềm Themis yêu cầu, tự sinh ra file Input và file Output, đưa vào cây thư mục trên.

(Hướng dẫn sử dụng Tiện ích và link download đã được tôi lưu vào đĩa CD nập kèm SKKN này)

b) Ưu điểm của Tiện ích sinh Test tự động

Ưu điểm lớn nhất khi sử dụng Tiện ích này vào việc xây dựng Test chấm là giúp giáo viên tiết kiệm thời gian, công sức. Giả sử cần tạo một bộ test chấm gồm 20 Test. Thay vì phải lặp đi lặp lại các công việc tạo thư mục Test, tạo tệp Input, chạy chương trình tạo tệp Output, di chuyển tệp Input và tệp Output vào thư mục con theo đúng cấu trúc 20 lần thì nhờ sử dụng Tiện ích Sinh Test tự động, tôi không cần phải tạo thư mục con mà chỉ cần chạy chương trình Tiện ích một lần duy nhất.

Tuy nhiên, để có thể tạo ra được các bộ test chấm hay, đánh giá được tương đối chính xác mức độ hoàn thiện bài làm của học sinh, thì giáo viên cần vận dụng linh hoạt Tiện ích, đặc biệt trong bước thay đổi code nguồn của tệp sinh Input. Dưới đây là những kinh nghiệm của tôi trong việc sử dụng *Tiện ích Sinh test tự động* vào việc xây dựng bộ Test chấm.

2.3.4 Vận dụng linh hoạt tiện ích sinh Test tự động để xây dựng bộ test chấm

a) Cách làm chung

B1: Viết code đáp án: Vì code này được sử dụng để sinh ra file output cho Test nên yêu cầu code phải đảm bảo chính xác, xét được các trường hợp input chứa dữ liệu có giá trị đặc biệt, dữ liệu có giá trị lớn theo yêu cầu của bài toán.

B2: Tạo file sinh Input: Để tạo ra các input đa dạng, có khả năng đánh giá tương đối mức độ hoàn thiện bài làm của học sinh, tôi không chỉ tạo một file sinh Input, mà tùy theo yêu cầu của đề tôi thường tạo ra nhiều file sinh Input khác nhau.

- Ban đầu tôi tạo chương trình sinh Input với mục đích tạo ra được các test *kiểm thử được độ chính xác* bài làm của học sinh. Bao gồm các trường hợp biên, cực hạn hoặc các trường hợp cho kết quả không tuân theo thuật giải, làm thay đổi đáng kể một vài yếu tố liên quan đến thời gian, bộ nhớ của máy tính. Theo kinh nghiệm của tôi thì đây là những test dễ sai nhất.

- Tiếp theo tôi tạo chương trình sinh test với mục đích tạo ta được các test có khả năng kiểm tra sức chịu đựng của chương trình. Đây thường là những test có dữ liệu lớn (tôi tạo dữ liệu Input theo cấp độ tăng dần về độ lớn)

B3: Chạy file Setup của tiện ích: nhập các thông số cho phù hợp để sinh ra input và output tương ứng với các trường hợp trên. Sau đó kích chọn Run.

Với cách làm này tôi luôn kiểm soát được dữ liệu input của bài toán, tôi chú ý tạo các test đặc biệt để đánh giá sát năng lực phân tích đề, làm bài của học sinh. Đồng thời nhờ việc tự động hóa của *Tiện ích sinh test* đã giúp tôi giảm đáng kể thời gian tạo test chấm.

b) Một số ví dụ

Ví dụ 1: Sinh Test cho bài tập SỐ LỚN THỨ NHÌ

Tệp ‘SOTHUNHI.INP’ bao gồm:

- Dòng đầu tiên chứa một số nguyên dương N ($N \leq 10^6$)
- Dòng tiếp theo chứa dãy A gồm N số nguyên ($|A_i| \leq 10^9$)

Yêu cầu: Chỉ ra số nguyên âm lớn thứ nhì, không xuất hiện trong dãy A .
Kết quả ghi vào tệp ‘SOTHUNHI.OUT’

Ví dụ:

SOTHUNHI.INP	SOTHUNHI.OUT
7	-2
3 -4 3 5 1 0 -3	

Bước 1: Viết code đáp án

(code đáp án đã được tôi lưu vào đĩa CD nộp kèm theo SKKN này)

Bước 2: Tạo file sinh Test

Phân tích:

Đối với bài tập VD1 ở trên, để xây dựng được bộ Test chấm có khả năng đánh giá tương đối mức độ hoàn thiện trong thuật giải của bài làm học sinh, tôi đã chú ý tạo ra các Test mà Input có dữ liệu vào ở giới hạn cận dưới, giới hạn cận trên và các Input có giá trị đặc biệt. Cụ thể Input có dữ liệu trong một số trường hợp sau:

- Dãy gồm một phần tử có giá trị nguyên dương (Test1)
- Dãy gồm một phần tử có giá trị nguyên âm (Test 2)
- Dãy gồm 10^6 phần tử giá trị các phần tử lớn hơn -10^6 (Test 12)
- Dãy gồm N phần tử, trong dãy có phần tử có giá trị bé hơn -10^6 , trị tuyệt đối không quá 10^9 (Test 11)
- Còn lại tôi tạo các Test ngẫu nhiên, với giá trị dữ liệu tăng dần

Thực hiện:

Test 1: Input gồm số $N = 1$ và một số nguyên dương

{Khi đó số âm lớn thứ nhất là số -1; như vậy số âm lớn thứ nhì là số -2
Nên Output cần đưa ra là: -2}

Để tạo tệp sinh input 1 như trên, tôi chỉ cần sửa lại code nguồn của tác giả từ đoạn try đến đoạn finally như sau:

try

Writeln(f, '1'); write(f,'5');

Finally

Sau đó lưu lại với tên R_SOTHUNHI_TEST1.PAS.

Test 2: Input gồm số N = 1 và một số nguyên âm

Đoạn lệnh được sửa lại trong code nguồn của tác giả (từ try đến đoạn finally) như sau:

Try

Writeln(f, '1'); write(f,'-2');

Finally

Test 3 đến Test 5: Tạo ra 3 bộ test mà Input gồm số N ngẫu nhiên có giá trị từ 1 đến 100; và dãy N phần tử có giá trị nguyên từ -10⁴ đến 10⁴

Để tạo ra số N có giá trị ngẫu nhiên tôi sử dụng hàm Random, (chú ý phải khai báo thư viện chuẩn Crt và có dòng lệnh Randomize bên trên). Do hàm Random(100) chỉ cho phép tạo số nguyên ngẫu nhiên có giá trị từ 0 đến 100 nên để tạo số nguyên có giá trị từ 1 đến 100 tôi phải viết: 1+ Random(100-1)

Tương tự, để tạo các phần tử trong dãy có giá trị nguyên từ -10⁴ đến 10⁴ tôi viết random(10000)-random(10000)

Đoạn lệnh được sửa lại trong chương trình code nguồn để Sinh Input như sau:

try

N:=1+random(100-1); Writeln(f, N);

For i:=1 to N do begin k:=random(10000)-random(10000);

write(f,k, ' ');

end;

Finally

Test 6 đến test 8: Tạo ra 3 bộ test mà Input gồm số N ngẫu nhiên có giá trị từ 100 đến 1000; và dãy N phần tử có giá trị nguyên từ -10⁶ đến 10⁶

Đoạn lệnh được sửa lại trong chương trình code nguồn để Sinh Input như sau:

try

N:=100+random(1000-100); Writeln(f, N);

For i:=1 to N do begin

k:=random(1000000)-random(1000000);

write(f,k, ' ');

end;

finally

Test 9: Tạo bộ test có Input gồm số $N = 10^4$ và dãy gồm 10^4 phần tử trong đó có $10^4 - 3$ phần tử đầu có giá trị tăng dần từ -10^4 đến -4 ; 3 phần tử cuối là $-2 -2 -2$ (số -2 xuất hiện 3 lần) {Khi đó Output cần đưa ra là: -3 }

try

$N:=10000;$

Writeln(f, N);

For i:=-10000 to -4 do write(f,i, '');

write(f,-2,'',-2,'',-2);

finally

Test 10: Tạo bộ test có Input gồm số $N = 10^5$ và dãy gồm 10^5 phần tử trong đó có $10^4 - 1$ phần tử đầu có giá trị giảm dần từ -2 đến -10^5 ; phần tử cuối là -2 { khi đó Output cần đưa ra là: -100001 }

try

$N:=100000; Writeln(f, N);$

For i:=-2 downto -100000 do write(f,i, '');

write(f,-2);

finally

Test 11: Tạo bộ test có Input gồm số $N = 10^6$ và dãy gồm 10^6 phần tử có giá trị giảm dần từ -1 đến -10^6 {Output cần đưa ra là: -1000002 }

try

$N:=1000000; Writeln(f, N);$

For i:=-1 downto -1000000 do write(f,i, '');

Finally

Test 12: Tạo bộ test có Input gồm số $N = 10^6$ và dãy gồm 10^6 phần tử trong đó có $10^6 - 3$ phần tử đầu có giá trị tăng dần từ $-10^6 + 3$ đến -1 ; ba phần tử cuối là $-10^9, -10^7, -10^7 - 19$ (tôi cố tình chọn các phần tử có giá trị nhỏ hơn 10^6) {Output cần đưa ra là: -999999 }

try

$N:=1000000; Writeln(f, N);$

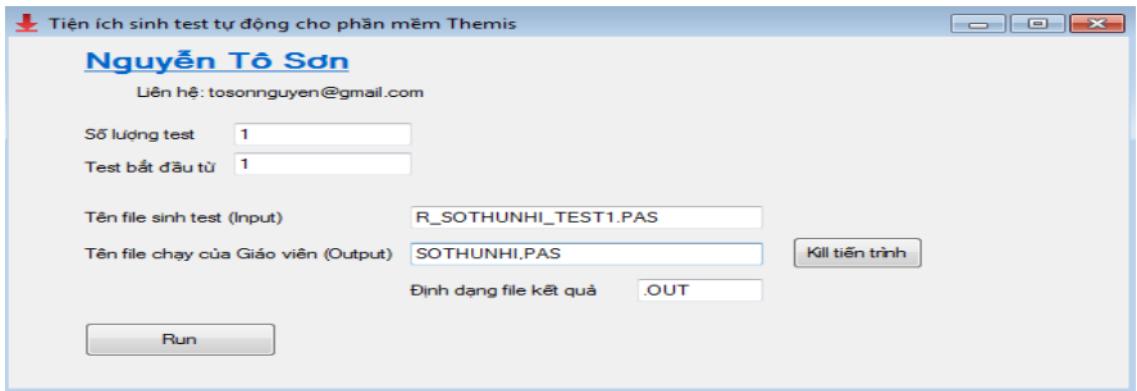
For i:=-1000000+3 to -1 do write(f,i, '');

Write(f,-1000000000,'',-100000000,'',-100000000-19);

Finally

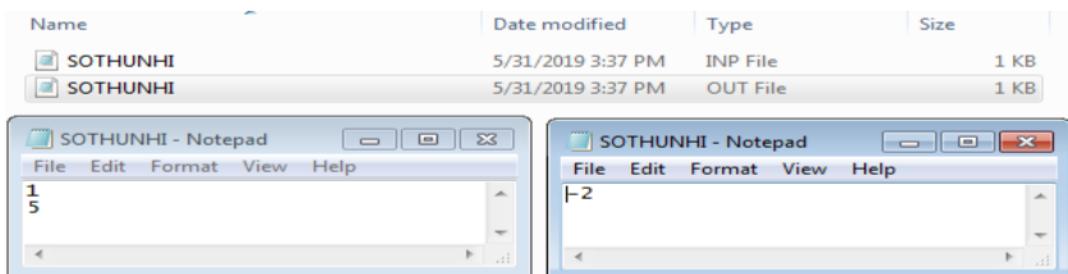
Sau mỗi thao tác trên tôi lưu lại tệp với một tên mới.

Bước 3: Chạy tiện ích Sinh test tự động, nhập các thông số tương ứng:

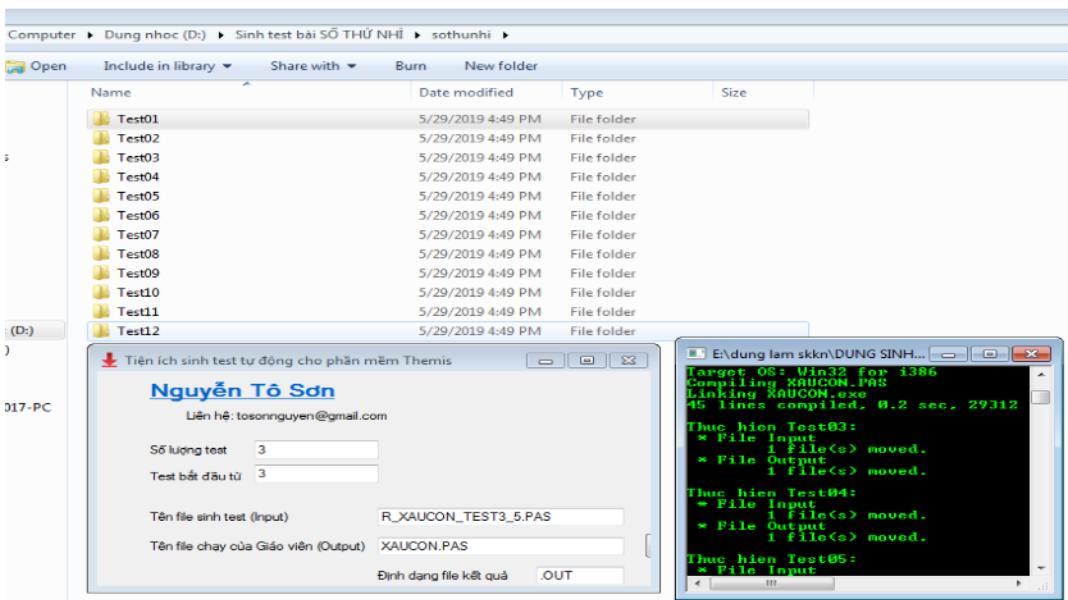


Lúc này, tiện ích sẽ liên kết, sinh ra tệp Input từ file R_SOTHUNHI_TEST1.PAS và sinh ra file output tương ứng từ tệp code đáp án SOTHUNHI.PAS mà tôi đã tạo trước đó.

Kết quả, tôi thu được bộ thư mục có tên là Test1 bao gồm Input, Output như sau:



Lặp lại thao tác trên, thay đổi thông số trong Tiện ích tôi sẽ thu được Input, Output cho các Test còn lại.



(Để minh họa, đáp án, các file sinh test và bộ test chấm hoàn chỉnh được tôi lưu vào đĩa CD nộp kèm theo SKKN này)

Như vậy, chỉ với việc sửa lại một đoạn ngắn trong code nguồn của tác giả, rồi chạy tiện ích tôi dễ dàng tạo ra được Bộ test chấm hoàn chỉnh, trong có nhiều Test có giá trị Input đặc biệt và các Test có giá trị ngẫu nhiên nhưng độ lớn và độ phức tạp tăng dần.

Ví dụ 2: Sinh Test cho bài tập XÂU CON

Cho xâu S độ dài n chỉ gồm các kí tự in hoa ‘A’ đến ‘Z’. Hãy tìm xâu con liên tiếp dài nhất trong xâu S, sao cho mỗi kí tự tham gia vào xâu con không quá k lần.

Yêu cầu: Chỉ ra độ dài của xâu con tìm được và vị trí của kí tự đầu tiên thuộc xâu con trong xâu S ban đầu. Nếu có nhiều cách chọn xâu con – chỉ ra xâu con đầu tiên trong xâu S và vị trí của kí tự đầu tiên thuộc xâu S ban đầu.

Input: File XAUCON.INP:

- Dòng đầu tiên chứa 2 số nguyên n và k ($1 \leq n \leq 100\ 000$, $1 \leq k \leq n$).
- Dòng thứ hai chứa xâu S.

Output: File XAUCON.OUT chỉ một dòng duy nhất ghi hai số nguyên là độ dài xâu con và vị trí kí tự đầu tiên của xâu con.

XAUCON.INP	XAUCON.OUT
5 1 HELLO	3 1

Tương tự như bài toán ở ví dụ 1, để thực hiện tạo bộ test chấm cho bài tập này tôi cũng đã thực hiện theo 3 bước trên, dưới đây tôi chỉ đề cập đến thao tác tạo file sinh test(B3), các bước còn lại làm tương tự như ví dụ 1.

Phân tích:

Để tạo bộ test chấm cho bài toán XÂU CON trên, tôi đã xét một số trường hợp dữ liệu có giá trị Input cực tiểu, giá trị cực đại, giá trị đặc biệt:

- Trường hợp xâu chỉ gồm một kí tự duy nhất (test1)
- Trường hợp xâu gồm N kí tự (N bé) và một số nguyên K=1 (Test2)
- Trường hợp xâu gồm N kí tự (N bé) và một số nguyên K > 1 (Test3)
- Trường hợp xâu gồm các kí tự giống hệt nhau, k>1, k<N (test4)
- Trường hợp xâu gồm 10^5 phần tử (Test 12)
- Còn lại tôi tạo các Test ngẫu nhiên, dữ liệu có giá trị tăng dần.

Code được sửa từ code mẫu của tác giả trong các trường hợp đó như sau:

Test1: Input chứa một số N ; số K =1 và một xâu gồm một kí tự.

Chương trình Sinh Input được sửa lại như sau:

procedure GenTest(const FName: string);

```

var
f: Text; i, k,n: integer;S:ANSISTRING;
begin
Randomize;
Assign(f, FName); rewrite(f);
try
  Writeln(f, '1', ' ', '1');  WRITE(F,'S');
finally
  close(f);
end;
end;

```

Test 2: Input chứa một số N; số K =1 và một xâu có độ dài N.

Đoạn lệnh được sửa lại như sau:

```

try
  S:='TOIYEUVIETNAM';k:=1;
  Writeln(f,LENGTH(S), ' ',K);  WRITE(F,S);
finally

```

Test 3: Input chứa một số N; số K =5 và một xâu có độ dài N.

Đoạn lệnh được sửa lại như sau:

```

try
  S:='TOIYEUVIETNAM';k:=5;
  Writeln(f,LENGTH(S), ' ',K);
  WRITE(F,S);
Finally

```

**Test 4: Input chứa một số N; số K =3 và một xâu gồm N kí tự giống
hết nhau.**

Đoạn lệnh được sửa lại như sau:

```

Try
  N:= 1+random(10); K:= 1+random(N-1);
  For i:= 1 to N do  WRITE(F, 'A');
finally

```

**Test 5 đến test 7: Input chứa một số N ngẫu nhiên $1 \leq N \leq 100$; số K có
giá trị ngẫu nhiên $1 \leq K \leq N$; và một xâu có độ dài N chứa các kí tự ngẫu
nhiên từ A đến Z.**

Để tạo tệp sinh Test có xuất hiện các kí tự A...Z ngẫu nhiên, tôi đã thực hiện: trước hết tạo xâu st chứa tất cả các kí tự từ A đến Z, sau đó tôi dùng hàm Random để tạo số nguyên k ngẫu nhiên từ 1 đến length(st). In ra st[k] tôi sẽ thu được kí tự ngẫu nhiên từ A đến Z.

Chương trình Sinh Input sau khi được tôi sửa lại từ chương trình nguồn sinh test của tác giả như sau:

```
// Chuong trinh Tao cac file Input cua Ban giam khao
{$MODE OBJFPC}
program RandomTest;
const
Prefix = 'XAUCON.INP';
var st: string;
ch: char;
len: integer;
procedure GenTest(const FName: string);
var
f: Text; i, k, n: integer;
begin
Randomize;
Assign(f, FName); rewrite(f);
try
N:= Random(100-1)+ 1;k:=RANDOM(N-1)+1; {*}
WRITELn(f,N,' ',K);
for i:= 1 to N do
begin    k:=random(length(st)-1)+1;write(f,st[k]); end;
finally
close(f);
end;
end;
procedure GenAZ;
var
c: Char;
begin
for c := 'A' to 'Z' do
```

```

    GenTest(Prefix);
end;
begin
st:="";
for ch:= 'A' to 'Z' do st:= st + ch;
GenAZ;
end.

```

Test 8 đến test 11: Input chứa một số N ngẫu nhiên $10^2 \leq N \leq 10^4$; số K có giá trị ngẫu nhiên $1 \leq K \leq N$; và một xâu có độ dài N chứa các kí tự ngẫu nhiên từ A đến Z.

Chương trình tôi sử dụng tương tự như chương trình sinh test 3-5 đã xây dựng ở trên, chỉ thay đổi duy nhất dòng lệnh {*} thành: $N := Random(10000-100)+100; k := RANDOM(N-1)+1;$

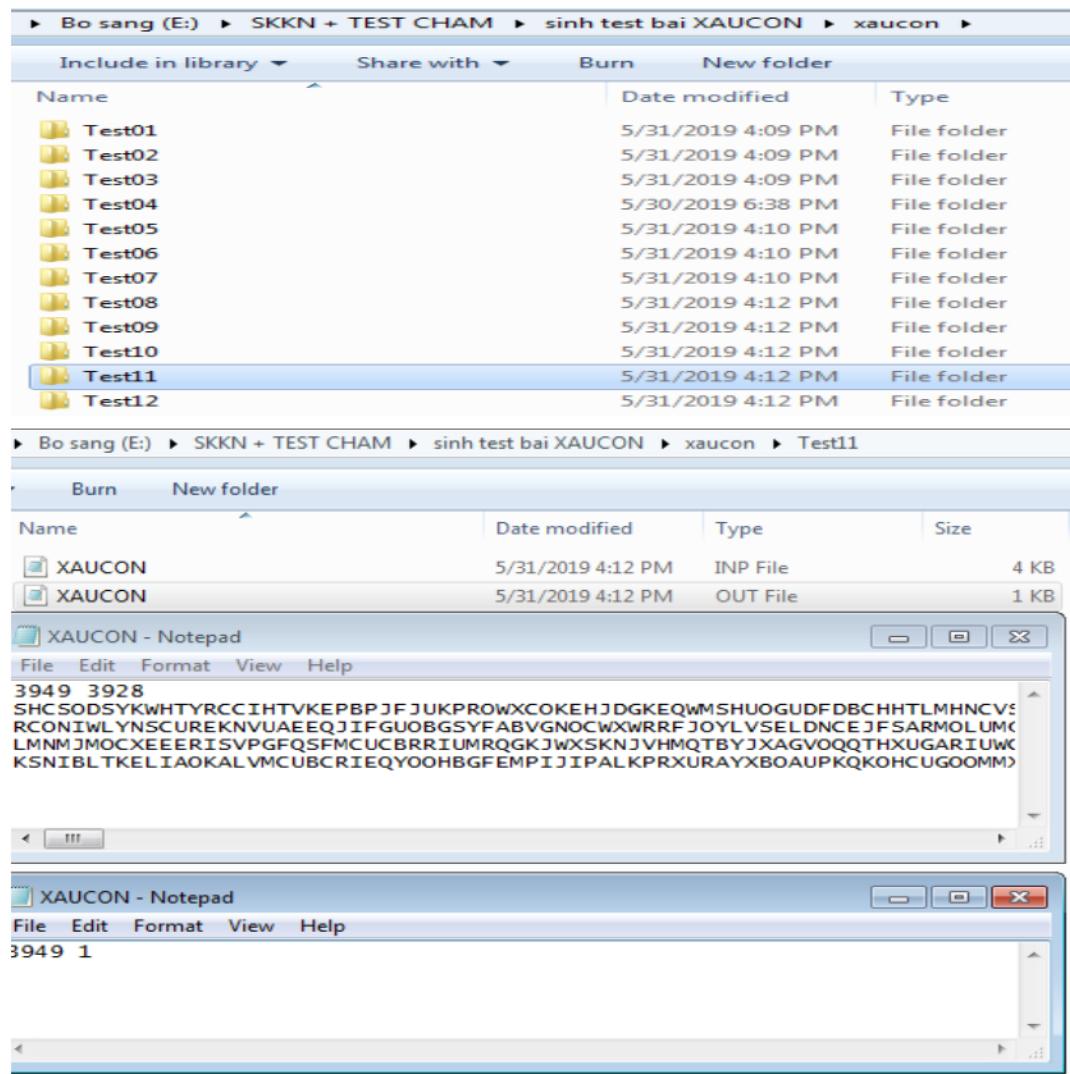
Test 12: Input chứa một số N = 10^5 ; số K có giá trị ngẫu nhiên $1 \leq K \leq N$; và một xâu có độ dài N chứa các kí tự ngẫu nhiên từ A đến Z.

Chương trình tôi sử dụng tương tự như chương trình sinh test 3-5 đã xây dựng ở trên, chỉ thay đổi duy nhất dòng lệnh {*} thành: $N := 100000; k := RANDOM(N-1)+1;$

Kết quả sau khi tạo được 5 file Sinh Input tương ứng với các trường hợp đã phân tích ở trên.

Name	Date modified	Type	Size
📁 xaucon	5/31/2019 4:12 PM	File folder	
📄 R_XAUCON_TEST1	5/31/2019 4:01 PM	PAS File	1 KB
📄 R_XAUCON_TEST2	5/31/2019 4:03 PM	PAS File	1 KB
📄 R_XAUCON_TEST3	5/31/2019 4:03 PM	PAS File	1 KB
📄 R_XAUCON_TEST4	5/31/2019 4:05 PM	PAS File	1 KB
📄 R_XAUCON_TEST5_7	5/31/2019 4:08 PM	PAS File	1 KB
📄 R_XAUCON_TEST8_11	5/31/2019 4:12 PM	PAS File	1 KB
📄 R_XAUCON_TEST9_10	5/29/2019 1:36 PM	PAS File	1 KB
📄 R_XAUCON_TEST12	5/31/2019 4:08 PM	PAS File	1 KB
📄 Tien ich sinh file test	1/2/2014 8:15 PM	Application	79 KB
📄 toson4	6/14/2014 9:29 PM	Application	5,496 KB
📄 XAUCON	5/27/2019 9:28 AM	PAS File	2 KB

Kết quả sau khi đã chạy Tiện ích Sinh Test tự động, tôi thu được cây thư mục test có nội dung:



(Để minh họa, đáp án, các file sinh test và bộ test chấm hoàn chỉnh được tôi lưu vào đĩa CD nộp kèm theo SKKN này)

Những lưu ý khi xây dựng chương trình sinh test:

- Tạo test nhỏ trước, test lớn sau
- Không viết quá nhiều test có input na ná nhau, nghĩa là các input này chỉ kiểm tra duy nhất một trường hợp của bài toán

Như vậy, nhờ sử dụng *Tiện ích sinh test tự động* đã giúp tôi tiết kiệm khá nhiều thời gian trong việc tạo Bộ Test chấm có số lượng Test theo mong muốn, input có dữ liệu có độ lớn tăng dần, để hướng đến đánh giá tính hoàn thiện dần của thuật giải trong bài làm của học sinh.

2.4 Hiệu quả của sáng kiến kinh nghiệm đối với hoạt động giáo dục, với bản thân, đồng nghiệp và nhà trường.

- **Đối với học sinh:** Việc tạo ra các bộ Test chấm và chấm bài cho học sinh trong quá trình bồi dưỡng và đánh giá chất lượng HSG giúp các em tự nhận ra sai lầm trong bài làm của mình. Từ đó rút ra được nhiều kinh nghiệm cho bản thân trong việc xử lí dữ liệu.

- Đối với bản thân và đồng nghiệp: Khi áp dụng Tiện ích sinh Test vào việc xây dựng Bộ Test chấm, tôi thấy việc tạo test chấm trở nên đơn giản hơn, nhanh chóng hơn, ít gây nhầm lẫn. Tôi có thể dành nhiều thời gian cho việc nghiên cứu đề, tối ưu thuật toán. Việc xây dựng được nhiều bộ test chấm giúp giáo viên chủ động hơn trong việc ôn luyện đội tuyển HSG. Giáo viên thông qua việc chấm bài cho học sinh sẽ biết được khả năng làm bài của từng em, từ đó có những định hướng nhất định trong kế hoạch ôn luyện.

Sau khi áp dụng đề tài và thường xuyên sử dụng các bộ Test chấm vào quá trình dạy bồi dưỡng HSG. Tôi đã kiểm tra đánh giá chất lượng HSG, kết quả các bài thi khảo sát như sau:

Kì Thi Khảo sát	Họ tên	Điểm câu 1	Điểm câu 2	Điểm câu 3	Điểm câu 4	Điểm câu 5	Tổng Điểm
Lần 3	Lê Thị Giang	5/6	4/5	3/4	1/3	0/2	13/20
	Vũ Văn Cường	5/6	3/5	3/4	0/3	0/2	11/20
Lần 4	Lê Thị Giang	6/6	0.5/5	4/4	3/3	0.5/2	14/20
	Vũ Văn Cường	6/6	3/5	3/4	1/3	1/2	14/20
Lần 5	Lê Thị Giang	6/6	4.5/5	2/4	1/3	1.5/2	15/20
	Vũ Văn Cường	6/6	5/5	2/4	1/3	0.5/2	14.5/20

So sánh kết quả giữa các bài thi khảo sát trên, và đối chiếu với kết quả bài thi khảo sát lần 1, lần 2 được tiến hành hồi đầu năm (mà tôi đã đưa ra ở phần thực trạng) dễ nhận thấy: Điểm kiểm tra của các lần sau thường cao hơn các lần trước đó, mặc dù đề có mức độ khó tăng dần. Đặc biệt học sinh thường có xu hướng trọn vẹn điểm trong các câu hỏi ở mức độ trung bình và trung bình khá (tức các em ít khi bị mất điểm trong các bài tập dạng này). Như vậy, học sinh đã dần biết phân tích kỹ đề, chú ý những điểm đặc biệt có trong đề, tránh để mất điểm trong các trường hợp dữ liệu có giá trị đặc biệt.

3. KẾT LUẬN VÀ KIẾN NGHỊ

3.1 Kết luận

Đề tài “**Sử dụng có hiệu quả tiện ích sinh test vào việc xây dựng các bộ Test chấm nhằm nâng cao chất lượng bồi dưỡng học sinh giỏi môn Tin học**” đã chỉ ra những hạn chế của phương pháp tạo bộ Test chấm theo cách thủ công và giới thiệu những ưu điểm, cách sử dụng Tiện ích sinh Test tự động để Tạo bộ Test một cách nhanh chóng, hiệu quả hơn.

Cần chú ý rằng, về cơ bản Tiện ích chỉ giúp ta tạo cây thư mục tự động nhờ việc đọc File sinh Input rồi tạo ra Input, đọc file Code đáp án rồi tạo ra Output tương ứng. Lắp Input, Output vừa tạo vào cây thư mục. Còn chương trình tạo Input như thế nào là giáo viên phải tự xây dựng dựa trên code mẫu của tác giả.

Điểm mới trong đề tài là đề tài không chỉ đưa ra những thao tác chung để tạo bộ Test chấm mà còn định hướng cách thức tạo ra các chương trình sinh Input sao cho chương trình có thể tạo ra các Test đa dạng, Test đặc biệt Test hướng đến tính hoàn thiện dần của thuật toán, và có khả năng đánh giá mức độ tối ưu trong bài làm của học sinh.

Hi vọng đề tài sẽ giúp ích cho các thầy cô giáo giảng dạy môn Tin học, đặc biệt là những giáo viên trẻ chủ động hơn trong công tác kiểm tra đánh giá học sinh lớp đội tuyển, từ đó có các kế hoạch kịp thời để bồi dưỡng, nâng cao chất lượng bồi dưỡng HSG.

Tiện ích sinh Test tự động được phát hành miễn phí, giao diện thân thiện và rất dễ sử dụng nên đề tài dễ dàng được áp dụng cho tất cả các giáo viên Tin học ở các trường THPT, và các em học sinh yêu thích bộ môn lập trình.

Trong quá trình nghiên cứu và triển khai đề tài này, tôi sử dụng ngôn ngữ lập trình PASCAL. Tuy nhiên đề tài hoàn toàn có thể được sử dụng để tiến hành với các code viết bằng ngôn ngữ lập trình C hoặc C++ bằng cách thay đổi phần cài đặt biến môi trường Path (xem thêm tại phần tài liệu tham khảo). Đối với những bài toán có nhiều đáp số, việc áp dụng đề tài có thể được sử dụng bằng cách viết thêm một trình chấm ngoài kết hợp với phần mềm chấm điểm tự động Themis, tiện ích sinh test sẽ giúp chuyển đổi định dạng từ .OUT sang .ANS theo quy định của Themis.

3.2 Kiến nghị

Đối với sở giáo dục và đào tạo: Hàng năm tổ chức các buổi tập huấn cho giáo viên cốt cán, giáo viên tham gia dạy đội tuyển để giáo viên có điều kiện giao lưu, học hỏi đồng nghiệp, nâng cao chuyên môn.

Đối với nhà trường: Tạo điều kiện cho giáo viên tham gia dạy đội tuyển. Duy trì công tác thi thử HSG giao lưu cụm các nhà trường, trong huyện.

Đối với tổ nhóm chuyên môn: Trong các buổi họp tổ - họp nhóm cần duy trì các hoạt động trao đổi chuyên môn, thảo luận các vấn đề khó để nâng cao

chuyên môn, phát huy hiệu quả của các buổi họp nhóm. Xây dựng ngân hàng đề thi, mỗi giáo viên sẽ sưu tầm các chuyên đề hay gửi vào ngân hàng đề của tổ nhóm, làm tư liệu tham khảo cho các thành viên của tổ nhóm.

Đối với giáo viên tham gia dạy đội tuyển: cần thường xuyên trao đổi để nâng cao chuyên môn

Trong quá trình thực hiện đề tài không tránh khỏi những thiết sót, hạn chế, rất mong các thầy cô giáo đóng góp ý kiến để sáng kiến kinh nghiệm của tôi được hoàn thiện hơn.

Tôi xin chân thành cảm ơn!

**XÁC NHẬN CỦA
THỦ TRƯỞNG ĐƠN VỊ**

Thanh Hóa, ngày 23 tháng 5 năm 2019

**Tôi xin cam đoan đây là SKKN
của mình viết, không sao chép nội dung
của người khác.**

Nguyễn Thị Dung