

روش‌های آزمایش نرم‌افزار

edu.mandegar@yahoo.com

دکتر عزیزاله محبی گرگری

درس مهندسی نرم‌افزار ۲

مفاهیم کلیدی

- آزمایش نرم افزار
- چه کسانی نرم افزار را آزمایش می کنند؟
- اصول آزمایش
- قابلیت آزمایش
- طراحی داده های آزمایش (*test case*)
- آزمایش جعبه سفید
- آزمایش جعبه سیاه

آزمایش نرم افزار

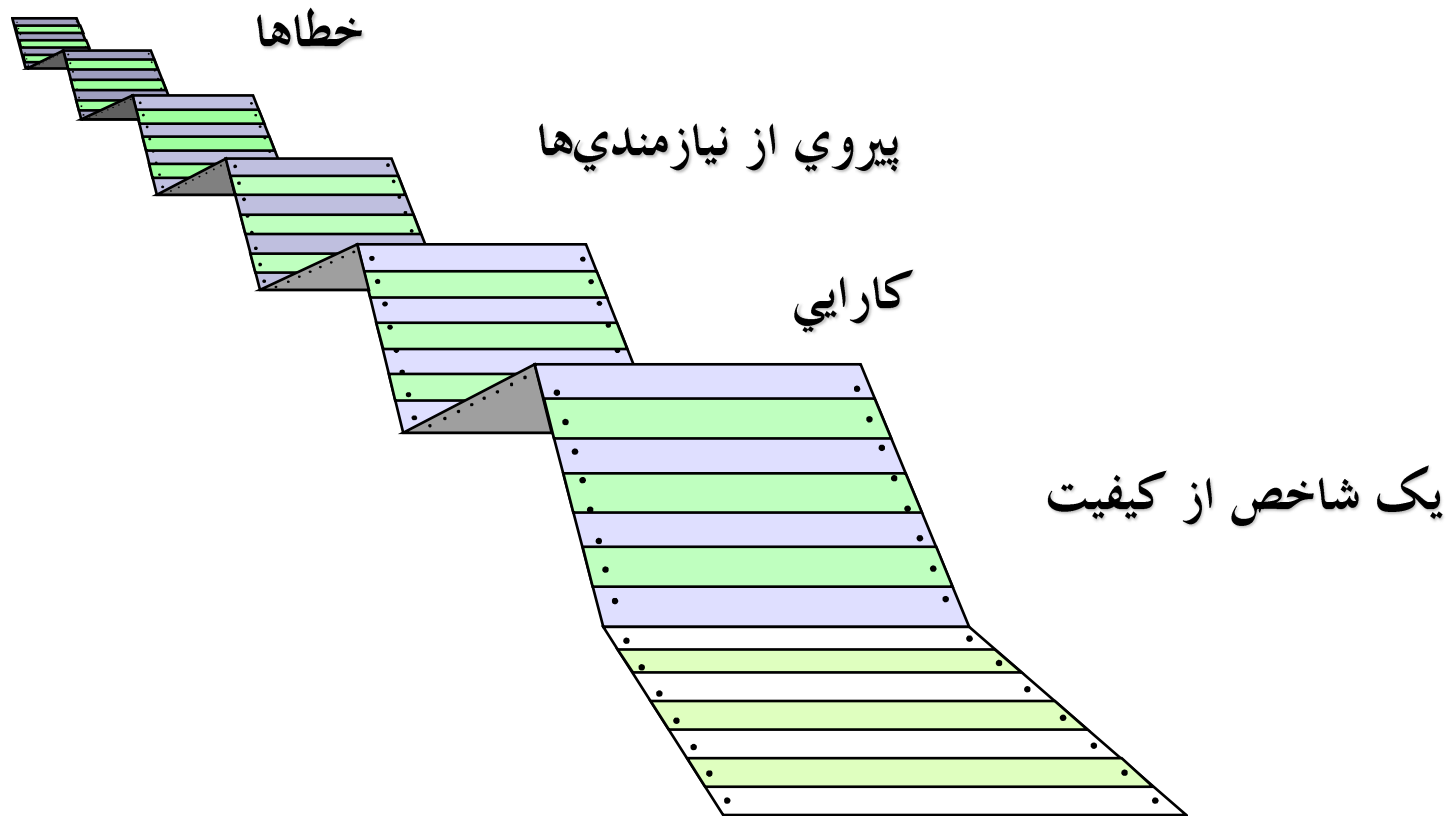
■ آزمایش نرم افزار

- فرآیند بازبینی نرم افزار با هدف یافتن خطاها قبل از تحویل به کاربر نهایی

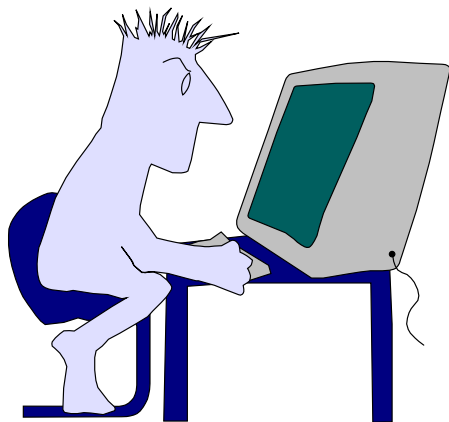
■ اهداف آزمایش نرم افزار (*Glen Myers*)

- آزمایش، فرآیند اجرای برنامه با هدف یافتن خطا است
- يك داده آزمایش (*test case*) خوب، نمونه‌ای است که با احتمال بالایی خطاها را بیابد
- آزمایش موفق، آزمایشی است که خطاهای تاکنون یافت نشده را بیابد

آزمایش نرم افزار چه چیزی را نشان می دهد؟



چه کسی نرم افزار را آزمایش می کند



آزمایش کننده مستقل

باید عملکرد سیستم را فراگیرد
اما تلاش می کند آن را از کار بیندازد
و هدفش کیفیت است



توسعه دهنده

سیستم را درک می کند،
اما آن را بتدریج آزمایش می کند
و هدفش تحویل نرم افزار است



و در نهایت کاربر نهایی سیستم را آزمایش می کند...

دکتر عزیزاله محبی گرگری

edu.manuegar@yahoo.com

اصول آزمایش نرم افزار (Davis)

- تمام آزمایشات باید به نیازمندی های مشتری قابل ردگیری باشند
- آزمایشات باید مدتی طولانی قبل از شروع، برنامه ریزی شوند
- اصل *Pareto* برای آزمایش نرم افزار بکار گرفته شود
- آزمایش باید با اجزاء کوچک شروع شود و به سمت آزمایش کل سیستم پیش رود
- آزمایش کامل و جامع امکان پذیر نیست
- به منظور تاثیرگذاری بیشتر، آزمایش باید توسط تیم مستقلی هدایت شود

قابلیت آزمایش

■ عملیاتی بودن (*Operability*)

- نرم افزار هرچه بهتر کار کند، با کارایی بالاتری می تواند آزمایش شود

■ قابلیت مشاهده (*Observability*)

- آنچه را می توانید ببینید، می توانید آزمایش کنید

■ قابلیت کنترل (*Controllability*)

- هر چه نرم افزار بیشتر قابل کنترل باشد، آزمایش بیشتر به طرز خودکار و بهینه قابل انجام است

■ پایداری (*Stability*)

- هر چه تغییرات کمتر باشد، انحراف از آزمایش کمتر است

قابلیت آزمایش (ادامه)

■ سادگی (*Simplicity*)

- هر چه موارد برای آزمایش کمتر باشد، آزمایش با سرعت بیشتری انجام می‌گیرد

■ قابلیت تجزیه‌پذیری (*Decomposability*)

- با کنترل نمودن محدوده آزمایش، با سرعت بیشتری مسائل تجزیه می‌شوند و آزمایشات هوشمندانه‌تری انجام می‌شود

■ قابلیت فهم (*Understandability*)

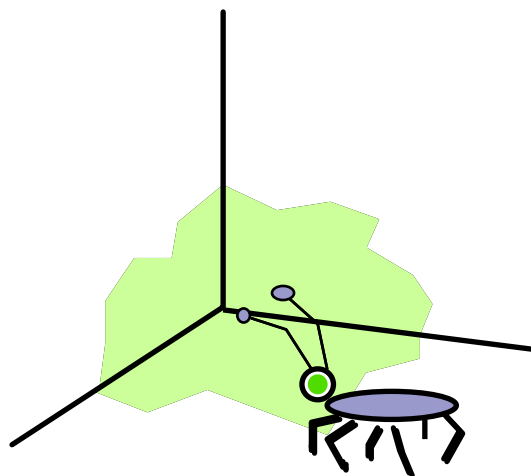
- هر چه اطلاعات بیشتری در اختیار داشته باشد، آزمایش هوشمندانه‌تری انجام می‌شود

آزمایش خوب

- با احتمال بالا، خطاها را نشان می‌دهد
- تکراری نیست
- جامعیت خوب و مناسب دارد
- نه بسیار ساده و نه بسیار پیچیده است



طراحی داده‌های آزمایش (*Test case*)

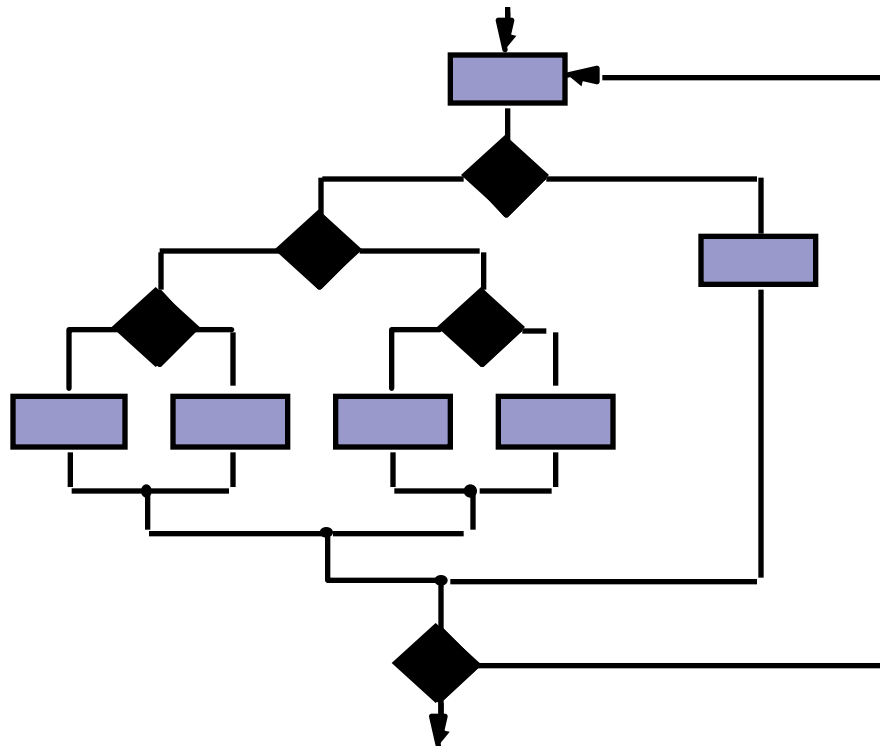


اشکالات در گوشه‌ها کمین می‌کنند و
در مرزها جمع می‌شوند

Boris Beizer

- هدف برای نمایش خطاها
- ملاک با روشی کامل
- اجبار با حداقل تلاش و زمان

آزمایش کامل

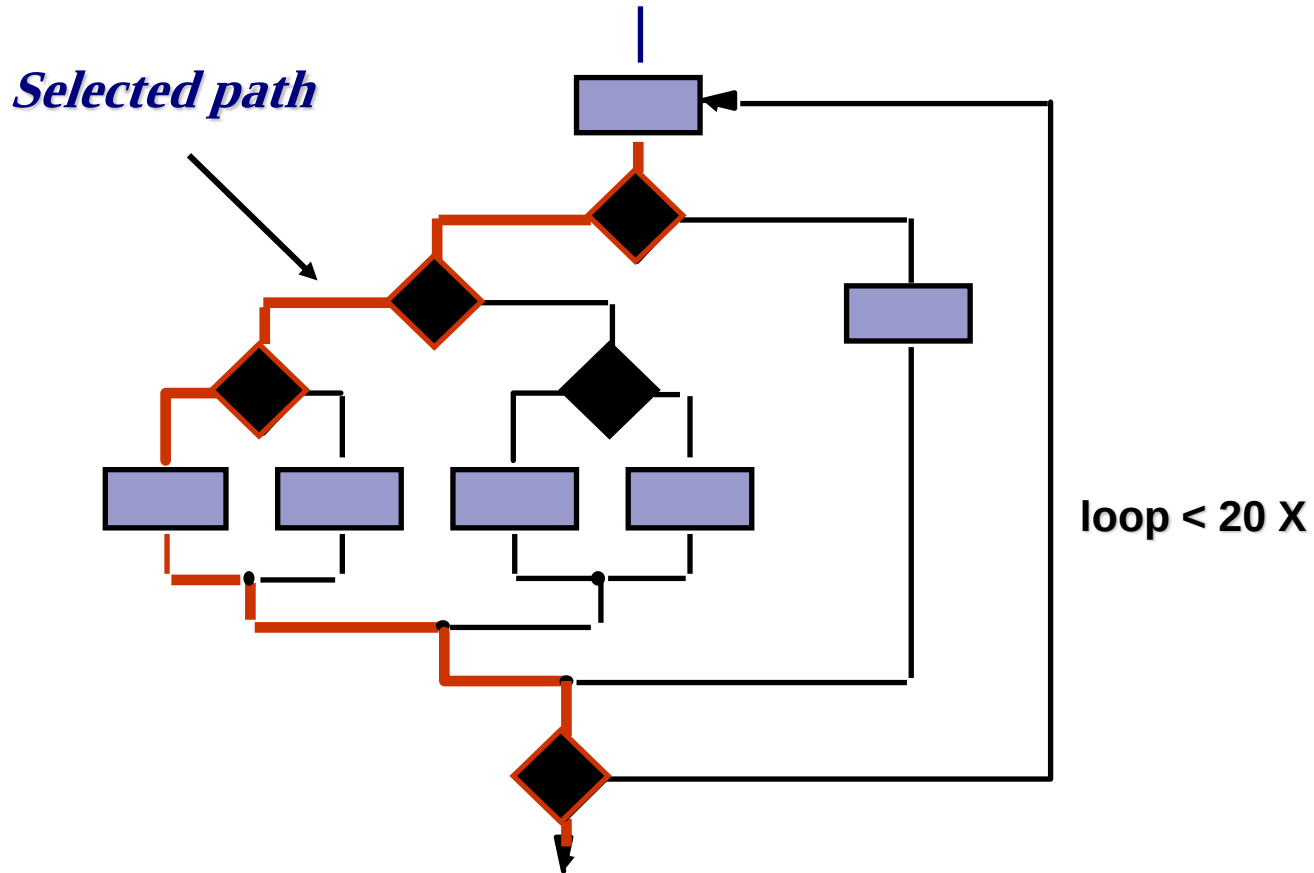


loop < 20 X



- برای آزمایش کامل یک برنامه ۱۰۰ خطی دارای دو حلقه تودرتو و چهار شرط که یک تا ۲۰ بار اجرا می‌شوند تقریباً به 10^{14} مسیر داریم که آزمایش کامل آنها ۳۱۷۰ سال طول می‌کشد!

آزمایش انتخابی



روش‌های آزمایش انتخابی نرم‌افزار



■ جعبه سیاه

• با دانستن تابع خاصی که يك محصول براي انجام آن طراحی شده

■ جعبه سفید

• با دانستن عملکرد داخلی محصول

استخراج مسیرهای پایه

- مقدار محاسبه شده برای پیچیدگی دورانی، تعداد مسیرهای مستقل را در مجموعه پایه برنامه مشخص می کند
 - حد بالایی برای تعداد آزمایشاتی که باید برای اطمینان از اجرای حداقل یک بار هر یک از دستورات انجام شوند
 - مسیر مستقل؛ هر مسیری در برنامه که حداقل یک مجموعه جدید از دستورات پردازشی یا شرط جدیدی را بیان نماید
- پس از استخراج مسیرهای پایه، داده آزمایش برای هر مسیر ایجاد می شود

نمونه از استخراج مسیرهای پایه

Example PDL

```

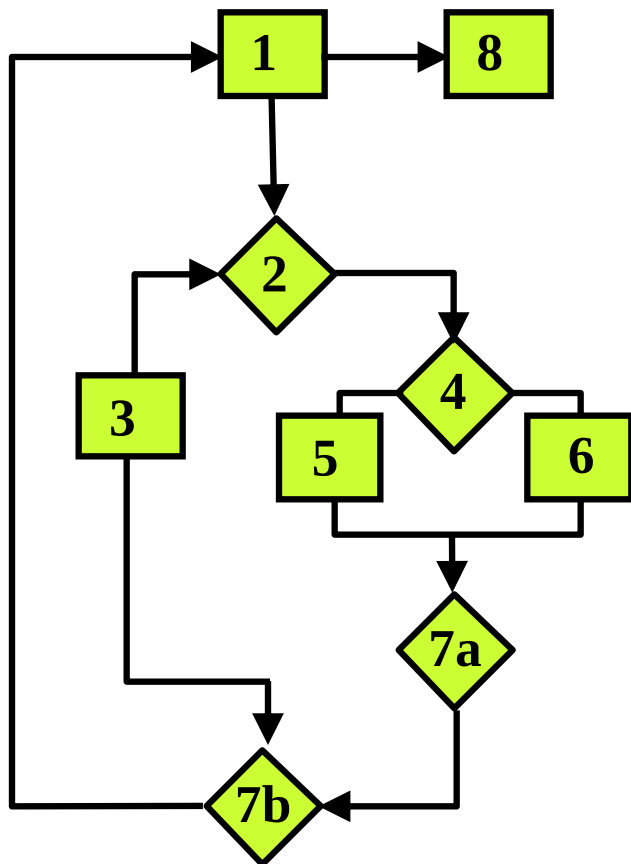
procedure sort
1 :   do while records remain
      read record ;
2 :   if record field1 = 0
3 :       then process record ;
           store in buffer ;
           increment counter ;
4 :       elseif record field2 = 0
5 :           then reset counter ;
6 :           else process record ;
               store in file ;
7a:       endif
           endif
7b:   enddo
8 : end

```

نمونه از استخراج مسیرهای پایه (ادامه)

■ مقدار پیچیدگی دورانی

$$V(G) = 11 - 9 + 2 = 3 + 1 = 4$$



Path 1: 1,8

Path 2: 1,2,3,7b,1,8

Path 3: 1,2,4,6,7a,7b,1,8

Path 4: 1,2,4,5,7a,7b,1,8

نمونه دیگر از استخراج مسیرهای پایه

PROCEDURE average;

* This procedure computes the average of 100 or fewer numbers that lie between bounding values; it also computes the sum and the total number valid.

INTERFACE RETURNS average, total.input, total.valid;

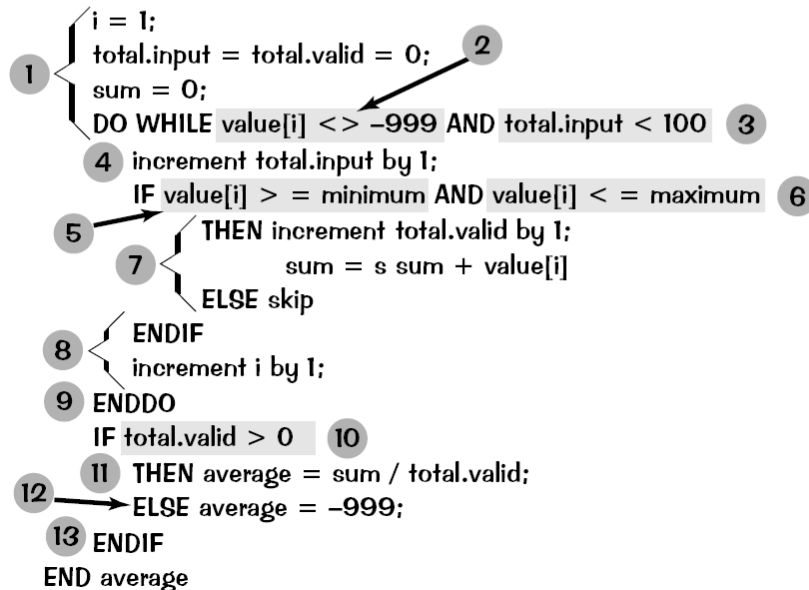
INTERFACE ACCEPTS value, minimum, maximum;

TYPE value[1:100] IS SCALAR ARRAY;

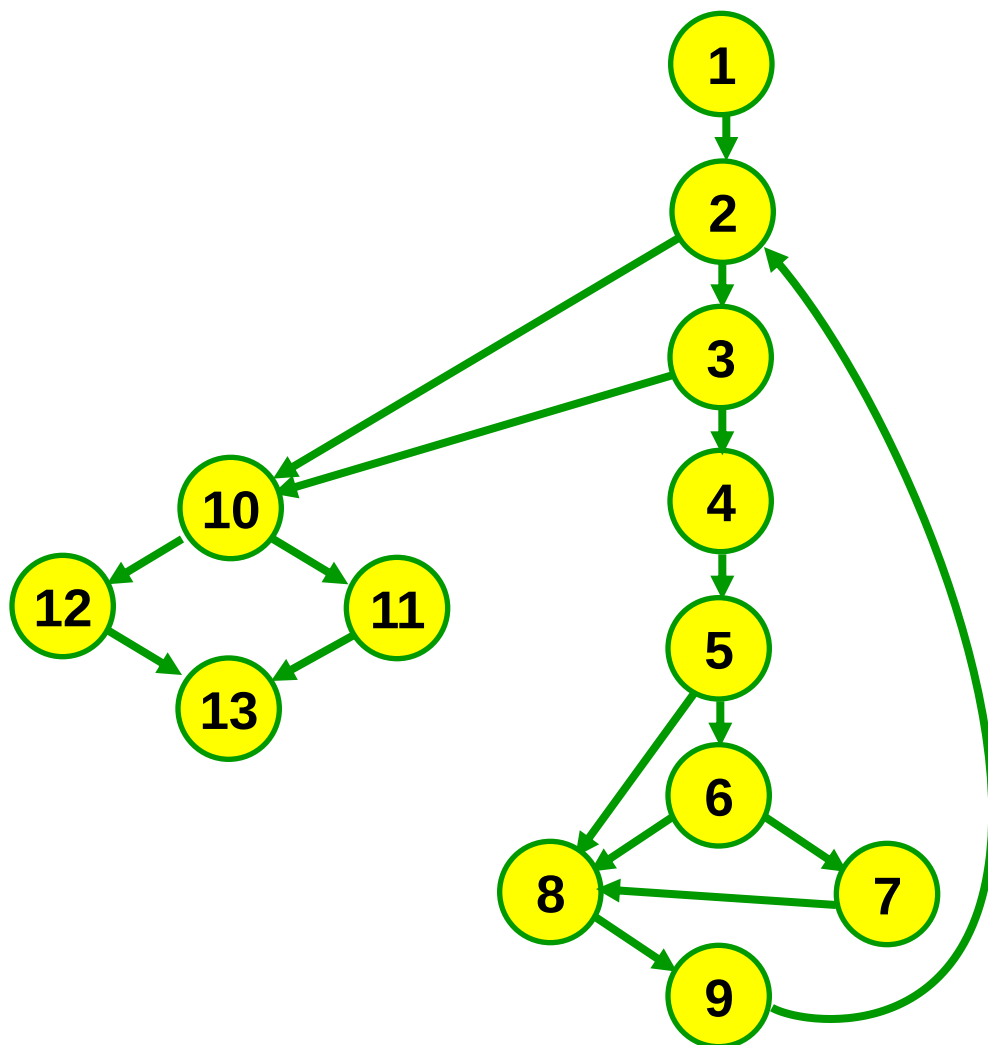
TYPE average, total.input, total.valid;

minimum, maximum, sum IS SCALAR;

TYPE i IS INTEGER;



نمونه دیگر از استخراج مسیرهای پایه (ادامه)



نمونه دیگر از استخراج مسیرهای پایه (ادامه)

$$V(G) = 6 \text{ regions} , V(G) = 17 \text{ edges} - 13 \text{ nodes} + 2 = 6 ,$$

$$V(G) = 5 \text{ predicates nodes} + 1 = 6$$

path 1: 1-2-10-11-13

path 2: 1-2-10-12-13

path 3: 1-2-3-10-11-13

path 4: 1-2-3-4-5-8-9-2-...

path 5: 1-2-3-4-5-6-8-9-2-...

path 6: 1-2-3-4-5-6-7-8-9-2-...

نمونه دیگر از استخراج مسیرهای پایه (ادامه)

Path 1 test case:

value(k) = valid input, where $k < i$ for $2 \leq i \leq 100$

value(i) = -999 where $2 \leq i \leq 100$

Expected results: Correct average based on k values and proper totals.

Note: Path 1 cannot be tested stand-alone but must be tested as part of path 4, 5, and 6 tests.

Path 2 test case:

value(1) = -999

Expected results: Average = -999 ; other totals at initial values.

Path 3 test case:

Attempt to process 101 or more values.

First 100 values should be valid.

Expected results: Same as test case 1.

Path 4 test case:

value(i) = valid input where $i < 100$

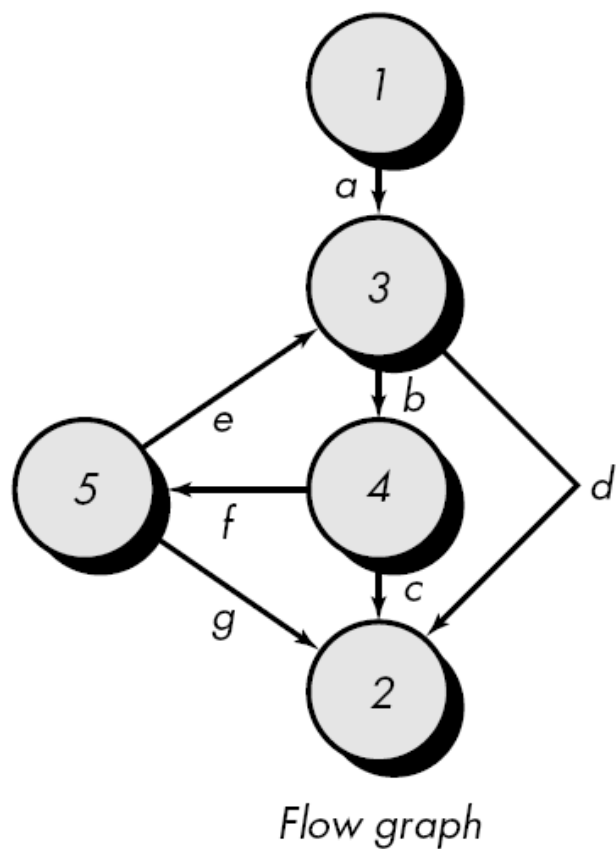
value(k) < minimum where $k < i$

Expected results: Correct average based on k values and proper totals.

ماتریس گراف (*Graph Metric*)

- ماتریس مربعی است که اندازه آن (یعنی تعداد سطرها و ستونها) برابر است با تعداد گرهها در گراف جریان
- هر سطر و ستون معادل يك گره مشخص است و ورودی‌های ماتریس معادل یال‌های بین گره‌ها می‌باشد
- با افزودن وزن یال به هر ورودی ماتریس، ماتریس گراف به ابزاری قدرتمند برای ارزیابی ساختار کنترل برنامه در ضمن آزمایش تبدیل شود

نمونه ماتریس گراف



Connected to node		1	2	3	4	5
Node		1	2	3	4	5
1				a		
2						
3			d		b	
4			c			f
5			g	e		

Graph matrix

نمونه ماتریس گراف (ادامه)

Connected to node		1	2	3	4	5	Connections
Node		1	2	3	4	5	
1				1			$1 - 1 = 0$
2							
3			1		1		$2 - 1 = 1$
4			1			1	$2 - 1 = 1$
5			1	1			$2 - 1 = 1$

Graph matrix

$\overline{3} + 1 = 4$

 Cyclomatic complexity

پایان