Exercise 2 – PSP Measurement

PSP/TSP[SM] Summer Faculty Workshop

PSP Summer Faculty Workshop

PSP Measurement Exercise

Overview

Exercise Overview	The exercise includes the following topics.

Prerequisites and References	Prerequisite: Read preface and chapters 1-2. Reference: Appendix C1

Exercise Objectives	After completing this exercise, you will • understand the basics of PSP measurement and the PSP0 process • understand how to use the time log, the defect log, and the plan summary

PSP0 Exercise Instructions	Look over the PSP0 scripts and then review the process forms. Then read the scenario for JD, a PSP student, doing assignment 1A. Using the data from this scenario, complete the time log, defect log, and plan summary for PSP0. If you are uncertain how to fill in the form, refer to the form instructions.

Table C10 PSP0 Process Script

	Purpose	To guide you in developing module-level programs
	Inputs Required	Problem description PSP0 project plan summary form Time and defect recording logs Defect type standard Stop watch (optional)
1	Planning	- Produce or obtain a requirements statement. - Estimate the required development time. - Enter the plan data in the project plan summary form. - Complete the time log.
2	Development	- Design the program. - Implement the design. - Compile the program and fix and log all defects found. - Test the program and fix and log all defects found. - Complete the time recording log.
3	Postmortem	Complete the project plan summary form with actual time, defect, and size data.
	Exit Criteria	- A thoroughly tested program - Completed project plan summary with estimated and actual data - Completed defect and time logs

Table C11 PSP0 Planning Script

Phase No.	**Purpose**	To guide the PSP planning process
	Entry Criteria	Problem description Project Plan Summary form Time recording log
1	Program Requirements	- Produce or obtain a requirements statement for the program. - Ensure that the requirements statement is clear and unambiguous. - Resolve any questions.
2	Estimate resources	- Make your best estimate of the time required to develop this program. - Distribute the development time over the planned project phases.
	Exit criteria	A documented requirements statement A project plan summary with estimated development time data Completed time log

Table C12 PSP0 Development Script

	Purpose	To guide the development of small programs
	Entry Criteria	- Requirements statement - Project plan summary with planned development time - Time and defect recording logs - Defect type standard
1	Design	- Review the requirements and produce a design to meet them. - Record time in time log.
2	Code	- Implement the design. - Record any requirements or design defects found in the defect recording log. - Record time in time log.
3	Compile	- Compile the program until error free. - Fix all defects found. - Record defects in defect log. - Record time in time log.
4	Test	- Test until all tests run without error. - Fix all defects found. - Record defects in defect log. - Record time in time log.
	Exit criteria	- A thoroughly tested program - Completed defect log - Completed time log

Table C13 PSP0 Postmortem Script

Phase No.	**Purpose**	To guide the PSP postmortem process
	Entry Criteria	- Problem description and requirements statement - Project plan summary with planned development time - Completed time log - Completed defect log - A tested and running program
1	Defects Injected	- Determine the defects injected in each PSP0 phase from the defect recording log. - Enter this number under Actual in the defects injected section of the project plan summary form.
2	Defects Removed	- Determine the defects removed in each PSP0 phase from the defect recording log. - Enter this number under Actual in the defects removed section of the project plan summary form.
3	Time	- Review the completed time recording log. - Enter the total time spent in each PSP0 phase in the Actual column of the project plan summary form.
	Exit criteria:	- A fully tested program - Completed project plan summary form - Completed defect and time logs

Table C17 Time Recording Log Instructions

Purpose	This form is for recording the time spent in each project phase. These data are used to complete the Project Plan Summary.
General	- Record all the time you spend on the project. - Record the time in minutes. - Be as accurate as possible. If you need additional space, use another copy of the form.
Header	Enter the following. - your name - today's date - the instructor's name - the number of the program - if you are working on a non-programming task, enter a job description in the Program # field.
Date	Enter the date when the entry is made.
Example	10/18/93
Start	Enter the time when you start working on a task.
Example	8:20
Stop	Enter the time when you stop working on that task.
Example	10:56
Interruption Time	Record any interruption time that was not spent on the task and the reason for the interruption. If you have several interruptions, enter their total time.
Example	37 - took a break
Delta Time	Enter the clock time you actually spent working on the task, less the interruption time.
Example	From 8:20 to 10:56, less 37 minutes or 119 minutes.
Phase	Enter the name or other designation of the phase or step being worked on.
Example	planning, code, test, etc.
Comments	Enter any other pertinent comments that might later remind you of any unusual circumstances regarding this activity.
Example	Had a compiler problem and had to get help.
Important	It is important to record all worked time. If you forget to record the starting, stopping, or interruption time for a task, promptly enter your best estimate for the time.

Table C16 Time Recording Log

Student	Niteshkumar S	Date	04/08/21
Instructor	Dr. Mohanraj N	Program #	Program1

Date	Start	Stop	Interruption Time	Delta Time	Phase	Comments
05/08 /21	10:00	11:00		60	Plan	
	11:00	11:30		30	Design	
	12:00	1:30	5	90	Code	Meeting with client
	2:00	2:50		50	Comp	
	3:00	3:30		30	Test	
	3:40	4:00		20	Postmo rtem	

Table C19 Defect Recording Log Instructions

Purpose	This form holds the data on each defect as you find and correct it. You use these data to complete the Project Plan Summary.
General	Record all review, compile, and test defects in this log. Record each defect separately and completely. If you need additional space, use another copy of the form.
Header	Enter the following. - your name - today's date - the instructor's name - the number of the program
Date	Enter the date when the defect was found.
Number	Enter the defect number. For each program, this should be a sequential number starting with 1 (or 001, etc.).
Type	Enter the defect type from the defect type list in Table C20 (also summarized in the top left corner of the log form). Use your best judgment in selecting which type applies.
Inject	Enter the phase during which this defect was injected. Use your best judgment.
Remove	Enter the phase during which the defect was removed. This would generally be the phase during which you found the defect.
Fix Time	Enter your best judgment of the time you took to fix the defect. This time can be determined by stopwatch or by judgment.
Fix Defect	If you injected this defect while fixing another defect, record the number of the improperly fixed defect. If you cannot identify the defect number, enter an X in the Fix Defect box.
Description	Write a succinct description of the defect that is clear enough to later remind you about the error and help you to remember why you made it.

Table C20 Defect Type Standard

DEFECT TYPES:

Type Number	Type Name	Description
10	Documentation	comments, messages
20	Syntax	spelling, punctuation, typos, instruction formats
30	Build, package	change management, library, version control
40	Assignment	declaration, duplicate names, scope, limits
50	Interface	procedure calls and references, I/O, user formats
60	Checking	error messages, inadequate checks
70	Data	structure, content
80	Function	logic, pointers, loops, recursion, computation, function defects
90	System	configuration, timing, memory
100	Environment	design, compile, test, or other support system problems

Table C18 Defect Recording Log

Defect Types

10 Documentation	**60 Checking**
20 Syntax	**70 Data**
30 Build, Package	**80 Function**
40 Assignment	**90 System**
50 Interface	**100 Environment**

Student	Niteshkumar S									Date	04/08/21
Instructor	Dr. Mohanraj N									Program #	Program 1
Date	Number		Type		Inject		Remove		Fix Time		Fix Defect
05/08/21	1		20		Code		Comp		1		
Description:	Missing ‘;’										
Date	Number		Type		Inject		Remove		Fix Time		Fix Defect
	2		20		Code		Comp		1		
Description:	Missing “)”										
Date	Number		Type		Inject		Remove		Fix Time		Fix Defect
	3,4		20		Code		Comp		4		
Description:	For loop indentation wrong										
Date	Number		Type		Inject		Remove		Fix Time		Fix Defect
	5		50		Code		Test		4		
Description:	Extra “ ” characters present in output										
Date	Number		Type		Inject		Remove		Fix Time		Fix Defect
	6		80		Design		Test		8		
Description:	For loop runs for n-1 times instead of n										
Date	Number		Type		Inject		Remove		Fix Time		Fix Defect
	7		10		Code		Test		1		
Description:	Function input output format not present in comments										
Date	Number		Type		Inject		Remove		Fix Time		Fix Defect
	8		80		Design		Test		12		
Description:	Function logic is wrong										
Date	Number		Type		Inject		Remove		Fix Time		Fix Defect
Description:											

Table C18 Defect Recording Log

Defect Types

10 Documentation	**60 Checking**
20 Syntax	**70 Data**
30 Build, Package	**80 Function**
40 Assignment	**90 System**
50 Interface 100 Environment	

Student		Date	
Instructor		Program #	

Date		Number		Type		Inject		Remove		Fix Time		Fix Defect
Description:												
Date		Number		Type		Inject		Remove		Fix Time		Fix Defect
Description:												
Date		Number		Type		Inject		Remove		Fix Time		Fix Defect
Description:												
Date		Number		Type		Inject		Remove		Fix Time		Fix Defect
Description:												
Date		Number		Type		Inject		Remove		Fix Time		Fix Defect
Description:												
Date		Number		Type		Inject		Remove		Fix Time		Fix Defect
Description:												
Date		Number		Type		Inject		Remove		Fix Time		Fix Defect
Description:												
Date		Number		Type		Inject		Remove		Fix Time		Fix Defect
Description:												

Table C15 PSP0 Project Plan Summary Instructions

Purpose	This form holds the estimated and actual project data in a convenient and readily retrievable form.
Header	Enter the following. - your name and today's date - the program name and number - the instructor's name - the language you used to write the program
Time in Phase	- Under Plan, enter your original estimate of the total development time. - Under Actual, enter the actual time in minutes spent in each development phase. - Under To Date, enter the sum of the actual time and the To Date time from your most recently developed program. - Under To Date %, enter the % of To Date time in each phase.
Defects Injected	- Under Actual, enter the number of defects injected in each phase. - Under To Date enter the sum of the actual numbers of defects injected in each phase and the To Date values from the most recently developed program. - Under To Date %, enter the % of the To Date defects injected by phase.
Defects Removed	- Under Actual, enter the numbers of defects removed in each phase. - Under To Date, enter the sum of the actual number of defects removed in each phase and the To Date value from the most recently developed program. - Under To Date %, enter the % of the To Date defects removed by phase. - After development, record any defects later found during program use, reuse, or modification.

Table C14 PSP0 Project Plan Summary

Student	Niteshkumar S	Date	04/08/21
Program	Sample Program	Program #	Program 1
Instructor	Dr. Mohanraj N	Language	C++

Time in Phase (min.)	Plan	**Actual**	**To Date**	**To Date %**
Planning		60	60	23.5
Design		20	20	7.8
Code		82	82	32.3
Compile		50	50	19.6
Test		30	30	11.8
Postmortem		13	13	5.0
Total		255	255	100
Defects Injected		**Actual**	**To Date**	**To Date %**
Planning		0	0	0
Design		0	0	0
Code		5	5	83
Compile		1	1	17
Test		0	0	0
Total Development		6	6	100
Defects Removed		**Actual**	**To Date**	**To Date %**
Planning		0	0	0
Design		0	0	0
Code		0	0	0
Compile		2	2	33
Test		4	4	67
Total Development		6	6	100
After Development				

JD Scenario for Assignment 1A	_____ Part 1 _____ JD begins work on assignment 1A [8:00] by reviewing the requirements in the assignment package, including the test requirements, to be sure he understands them. He copies the requirements to his note pad. Then, based on the data presented on past student performance and JD's feeling about his own performance, he estimates that this assignment will take three hours. He writes this estimate on his note pad [8:06]. _____ Part 2 _____ After taking a break for some coffee, JD starts to design the program [8:10]. He sketches out a diagram of the linked list structure, identifies the routines he'll need for handling the linked list and for computing the mean and standard deviation. JD moves on to coding [8:31]. While working on coding, JD is interrupted by a classmate who doesn't understand how to get started. JD spends 10 minutes explaining how to use the PSP0 process forms and then gets back to coding. JD finishes coding all the routines, checks to make sure he hasn't missed anything [9:44] and fetches a fresh cup of coffee before compiling. _____ Part 3 _____ JD compiles the program [9:56] and gets an error message, missing semicolon. Looking at the compiler output, JD sees where the missing semicolon belongs and fixes the source code [9:57]. JD recompiles the program and gets another error message, undeclared

	identifier [9:58]. Surprised, since he thought he declared this identifier, JD searches through the source code and discovers that the identifier he declared had an '_' in it and this one didn't. He fixes the error, then quickly scans the rest of the source code and finds two more places where he left out the '_' and also fixes them [10:01]. JD again recompiles the program and gets another error message, incorrect parameter type [10:02]. JD studies the code for a minute, sees the error and fixes the source code [10:03]. JD again recompiles the program and gets an error message at the end of the program, unmatched begin [10:05]. After reviewing the program logic for a few minutes, JD spots where the missing end belongs and fixes the source code [10:08]. JD recompiles the program and this time, there are no compile errors [10:09]. _____ Part 4 _____ JD loads the program and begins executing the first test case [10:10]. The program prompts JD for the input data file name and JD types it in, but nothing happens [10:11]. JD invokes the debugger, traces the program execution, and discovers it is in an infinite loop. He studies the source code for the loop and spots the problem–a pointer was not incremented within the loop [10:22]. JD corrects the source code, recompiles the program and begins executing the first test case again. This time, the program outputs some results, but the print format is wrong, so JD can't tell if they're correct [10:23]. JD fixes the print format [10:25] and retries the first test case [10:26]. The format is OK now, but the answers are wrong. JD reviews the program logic and looks at some variables with the debugger. After studying the code and the results, JD realizes his initial design of the standard deviation was flawed and it needs to be rewritten [10:43]. JD rewrites the routine and recompiles it [10:51]. There is one compile error – JD left out another semicolon, so he quickly corrects the defect and recompiles the program [10:52]. This time there are no errors. JD re-executes the first test case and this time, the results are good [10:54]. JD executes the next two test cases and both give the correct results [10:57]. _____ Part 5 _____ JD finds his Plan Summary form and begins filling it in [10:58]. It takes him 13 minutes to complete the Plan Summary.