K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

Batch: A2	**Roll No.: 1911027**
Experiment / assignment / tutorial No. 2	

TITLE: Project Metric estimations for Mini Project

AIM: To enable the students learn different techniques for performing software size and cost estimation

Expected Course outcome of Experiment:

CO: Understand the software development process and Estimate different types of resources for the given project.

Books/ Journals/ Websites referred:

1. Roger Pressman, "Software Engineering", sixth edition, Tata McGraw Hill.
2. http://sunset.usc.edu/csse/research/COCOMOII/cocomo_main.html
3. http://sunset.usc.edu/research/COCOMOII/expert_cocomo/expert_cocomo2000.html

Pre Lab/ Prior Concepts:

Software projects have tendency of going past their deadline, going over budget, or both. The problem lies in the estimation of the amount of effort required for the development of a project. The cost estimation is usually dependent upon the size estimate of the project, which may use lines of code or function points as metrics. There are several different techniques for performing software cost estimation, including expert judgement and algorithmic models. Estimation by expert judgement is a common way of estimating the effort required for a project. Unfortunately, this method of estimation does not emphasize re-estimation during the project life cycle, which is an important part of project tracking, because it allows the estimates to be improved during the project life cycle. The quality of a cost estimation model is not so much attributed to the initial estimate, but rather the speed at which the estimates converges to the actual cost of the project. COCOMO is a popular algorithmic model for cost estimation whose cost factors can be tailored to the individual development environment, which is important for the accuracy of the cost

estimates. More than one method of cost estimation should be done so that there is some comparison available for the estimates. This is especially important for unique projects. Cost estimation must be done more diligently throughout the project life cycle so that in the future there are fewer surprises and unforeseen delays in the release of a product.

Estimation of size and cost of the developing project is required for the following major decision situations

- Financial decisions involving a software development effort
- Setting project budgets and schedules as a basis for planning and control
- Deciding on or negotiating tradeoffs among software cost, schedule, functionality, performance or quality factors
- Making software cost and schedule risk management decisions
- Deciding which parts of a software system to develop, reuse, lease, or purchase
- Making legacy software inventory decisions: what parts to modify, phase out, outsource, etc
- Deciding how to implement a process improvement strategy, such as that provided in the SEI CMM

Defining Cost estimation:

Cost estimation can be defined as the approximate judgement of the costs for a project. Cost estimation will never be an exact science because there are too many variables involved in the calculation for a cost estimate, such as human, technical, environmental, and political. Furthermore, any process that involves a significant human factor can never be exact because humans are far too complex to be entirely predictable. Furthermore, software development for any fair-sized project will inevitably include a number of tasks that have complexities that are difficult to judge because of the complexity of software systems.

Cost estimation is usually measured in terms of effort. The most common metric used is person months or years (or man months or years). The effort is the amount of time for one person to work for a certain period of time. It is important that the specific characteristics of the development environment are taking into account when comparing the effort of two or more projects because no two development environments are the same. A clear example of differences in development environments are the amount of time people work in different countries; the typical workweek in North America is 40 hours per week, while in Europe the typical workweek is 35 hours per week. Thus, when comparing a project from North America with a project from Europe, a conversion factor would have to be used to all for an accurate comparison. Different variables can be used

for cost estimation, which leads to a difficulty when comparing projects if standard models or tools are not used. For example, a cost estimate can include factors from management, development (e.g., training, quality assurance), and other areas specific to an organization.

Estimator:

The people who do the cost estimates could be either directly or indirectly responsible for the implementation for a project, such as a developer or manager, respectively. Someone who has knowledge of the organization and previous projects could use an analogy-based approach to compare the current project with previous projects, which is a common method of estimation for small organizations and small projects. The historical data is often limited to the memory of the estimator. In this case, the estimator would need to be experienced and would likely have been with the company for awhile.

Some people believe it is better if the estimates are done by outsiders so that there is less chance of bias. It is true that people outside an organization will likely have to deal with fewer company politics than people within the organization. For example, the developer for a company may want to please the manager and so give an estimate that is overly-optimistic. The disadvantage of having an outside estimate is that the person would have less knowledge of the development environment, especially if the person is from outside the company. An empirical method of estimation would then be required, such as the Constructive Cost Model (COCOMO. Empirical methods of estimation can be used by all types of estimators. There may be some resistance to using an empirical method of estimation because there may be some question on whether a model could outperform an expert. People who are accurate estimators are rare in our experience, and so it is best to get the opinion of several people or tools.

Cost estimation using different COCOMO models:

Software Size Sizing Method: Source Lines of Code

	SLOC	% Design Modified	% Code Modified	% Integration Required	Assessment and Assimilation (0% - 8%)	Software Understanding (0% - 50%)	Unfamiliarity (0-1)
New	4000						
Reused		0	0				
Modified							

Software Scale Drivers

Precedentedness	Low	Architecture / Risk Resolution	Nominal	Process Maturity	Nominal
Development Flexibility	High	Team Cohesion	High		

Software Cost Drivers

Product		Personnel		Platform	
Required Software Reliability	Nominal	Analyst Capability	Nominal	Time Constraint	High
Data Base Size	High	Programmer Capability	Nominal	Storage Constraint	High
Product Complexity	High	Personnel Continuity	High	Platform Volatility	Nominal
Developed for Reusability	Nominal	Application Experience	Low	**Project**	
Documentation Match to Lifecycle Needs	Nominal	Platform Experience	Nominal	Use of Software Tools	Very High
		Language and Toolset Experience	High	Multisite Development	Low
				Required Development Schedule	Nominal

Maintenance On

Annual Change Size (ESLOC) 300 Maintenance Duration (Years) 0.8

Software Understanding (0%-50%) 35 Unfamiliarity (0-1) 0.2

Software Labor Rates

Cost per Person-Month (Dollars) 300

Calculate

Software Scale Drivers	Value	Justification
Precedentedness	Low	We have not implemented any project of this kind prior to this one, neither we have worked on a project with such complexity and this particular tech stack.
Architecture / Risk Resolution	Nominal	The Architecture/Risk Resolution is nominal since the product will be encompassing a GPS based tracking system and a face recognition system which will then proceed to mark attendance in the university/college database.
Process Maturity	Nominal	Due to low precedenteness in the project the extent of definiteness of the processes is nominal.
Development Flexibility	High	The development Flexibility will be high since the app will be dependent on the GPS and internet connection of the student and if there is low connectivity in particular areas the student may not be able to record his/her attendance
Team Cohesion	High	The team cohesion is bound to be high since all the members developing the product are under the same branch hence the time devoted towards the development of the product will be more.
Software Cost Drivers		
Product		
Required Software Reliability	Nominal	Is nominal as it is dependent on factors that may be changed or altered depending upon certain conditions. Considering different 4G Network providers, not all networks are available in SVU so that itself may pose a major issue.
Data Base Size	High	Since we are aiming at developing this system for KJSCE, recording attendance for 600 students from every year throughout the semester and for every lecture would require a large database.

Product Complexity	High	The product Complexity will be high since there is GPS based tracking system along with facial recognition system that increases the overall complexity of the product.
Developed for Reusability	Nominal	The idea of project allows it to be reused for any organisation wanting a smart attendance system with some minor shifts.
Documentation Match to Lifecycle Needs	Nominal	Will be nominal since we cannot cover all the aspects and have solution to problems pertaining to network issues etc.
Personnel		
Analyst Capability	Nominal	Our project is not focused or based on data as a main stream element although we collect data for face recognition which is a crucial element. Hence the value is nominal.
Programmer Capability	Nominal	We are still in the process of learning Flutter and Dart for apo development so as for now the programming capability is Nominal.
Personnel Continuity	High	We would be working on the project continuously during the span of about 6 months.
Application Experience	Low	Is low as we have not yet build any application or product on flutter.
Platform Experience	Nominal	Is nominal since we are very much familiar with the platform used to develop the application.
Language and Toolset Experience	Low	Is also low considering we are still in the learning stage.
Platform		
Time Constraint	High	The time constraint will be high since we are not yet familiar with the concepts of app development using flutter so it will take some time for us to get the hang of it.
Storage Constraint	High	The storage Constraint at the student level will be nominal but cannot be guaranteed considering there may be additional advancements or the current features may take up more storage than anticipated.

Platform Volatility	Nominal	The platform is not much volatile at this particular position in time but seeing the current advancements in the field of technology, we speculate that there might be a better, more efficient way of carrying out the task.
Project		
Use of Software Tools	Very High	As we are developing a system which marks attendance using a person's mobile phone so the use for a dedicated hardware device is discarded, as now a days every individual has his/her personal mobile phone. So this decreases our hardware cost to just a server depending on the requirements.
Multisite Development	Low	This value is low because the product does not demand a team on the client side. The developer team can handle both the development as well as the client communication.
Required Development Schedule	Nominal	Will be nominal since only a brief schedule will be shared with the customer or company to which the product is going to be delivered.
Maintenance		
Annual Change Size (ESLOC)	300	This is a pure guess as it would not be any estimated answer for this. This value is mainly determined by the situation at the time of maintenance.
Maintenance Duration (Years)	0.8	This would be a standard time for the maintenance.
Software Understanding (0%-50%)	35	Software must be understood fairly for the maintenance.
Unfamiliarity (0-1)	0.2	If we have understood the software completely there is always a possibility of introduction of a new thing whenever we enhances in the direction of development.
Software Labour Rates		

Cost per Person-Month (Dollars)	300	This value is taken considering that what we have thought of the project to be at the end. If we consider that the project is delivered on time with all necessary features then this value can be justified.

Results :

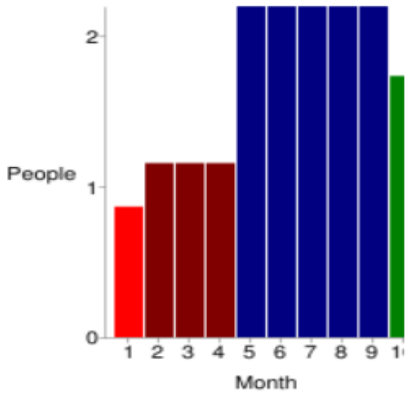
Results

Software Development (Elaboration and Construction)

Effort = 15.9 Person-months
Schedule = 8.8 Months
Cost = $4765

Total Equivalent Size = 4000 SLOC
Effort Adjustment Factor (EAF) = 1.19

Staffing Profile



Acquisition Phase Distribution

Phase	Effort (Person-months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	1.0	1.1	0.9	$286
Elaboration	3.8	3.3	1.2	$1144
Construction	12.1	5.5	2.2	$3622
Transition	1.9	1.1	1.7	$572

Software Effort Distribution for RUP/MBASE (Person-Months)

Phase/Activity	Inception	Elaboration	Construction	Transition
Management	0.1	0.5	1.2	0.3
Environment/CM	0.1	0.3	0.6	0.1
Requirements	0.4	0.7	1.0	0.1
Design	0.2	1.4	1.9	0.1
Implementation	0.1	0.5	4.1	0.4
Assessment	0.1	0.4	2.9	0.5
Deployment	0.0	0.1	0.4	0.6

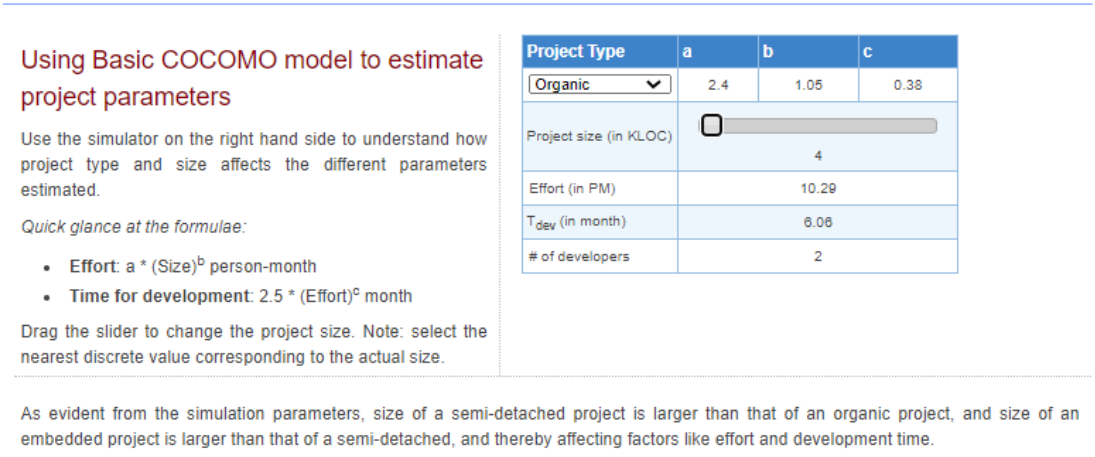
Maintenance
Annual Maintenance Effort = 1.0 Person-Months
Annual Maintenance Cost = $304
Total Maintenance Cost = $243

Virtual Lab :

Introduction :

After gathering the entire requirements specific to software project usually we need to think about different solution strategy for the project. Expert business analysts are analysing their benefits and as well as their shortcomings by means of cost, time and resources require to develop it. In this experiment, we will learn how to estimate cost, effort and duration for a software project, and then select one solution approach which will be found suitable to fulfil the organizational goal.

Simulation :



As evident from the simulation parameters, size of a semi-detached project is larger than that of an organic project, and size of an embedded project is larger than that of a semi-detached, and thereby affecting factors like effort and development time.

Exercises :

Considering your immense expertise in software development, The Absolute Beginners Inc. has recently allotted you a mega project. The goal of the project is to create a database of all Hindi films released since 2000. The software would allow one to generate a list of top ten hit films, top ten flop films, best comedy films, and so on. Using your prior experience you have decided the approximate sizes of each module of the software as follows:

- Data entry (0.9 KDSI)
- Data update (0.7 KDSI)
- Query (0.9 KDSI)
- Report generation and display (2 KDSI)

Also take into consideration the following cost drivers with their ratings:

- Storage constraints (Low)
- Experience in developing similar software (High)
- Programming capabilities of the developers (High)
- Application of software engineering methods (High)
- Use of software tools (High)

(All other cost drivers have nominal rating).

Now answer the following:

- Applying intermediate COCOMO estimate the minimum size of the team you would require to develop this system
- Assuming that your client would pay Rs. 50,000 per month of development, how much would be the likely billing?

Learning Objectives:

1. Identify type of a project as per COCOMO
2. Prepare an estimate of required effort and cost

Limitations: Values presented here are arbitrary and doesn't relate to real life

Note: The above example has been adapted from COCOMO (Constructive Cost Model), Seminar on Software Cost Estimation WS 2002 / 2003, presented by Nancy Merlo – Schett.

Submit

Project Type	a	b	c
Organic	2.4	1.05	0.38
Project size (in KDSI)	4.5		
Effort (in PM)	11.64		
T_{dev} (in month)	6.35		
Effort Adjustment Factor (EAF)	0.7		
$Effort\|_{corrected}$ (in PM)	8.15		
$T_{dev}\|_{corrected}$ (in month)	5.55		
# of developers	2		
Calculate			

K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

Identify the unique operators and operands from the following snippet of code:

```
01 int
02 main(int argc, char **argv)
03 {
04     int x = 10;
05     int y = 20;
06     int sum;
07
08     sum = x + y;
09
10     printf("Sum of %d and %d is: %d\n", x, y, sum);
11
12     return 0;
13 }
```

Submit

Note: In cases where you've to enter multiple values (for example, listing the operators from a code snippet), please separate them with a comma

Operators	☑ int ☑ main ☐ argc ☑ char ☑ * ☐ argv ☑ () ☑ {} ☐ x ☑ = ☐ 10 ☑ ; ☐ y ☐ 20 ☐ sum ☑ + ☑ printf ☐ "Sum of %d and %d is: %d " ☑ , ☑ return ☐ 0
Operands	☐ int ☐ main ☑ argc ☐ char ☐ * ☑ argv ☐ () ☐ {} ☑ x ☐ = ☑ 10 ☐ ; ☑ y ☑ 20 ☑ sum ☐ + ☐ printf ☑ "Sum of %d and %d is: %d " ☐ , ☐ return ☑ 0
Check	

Result

Excellent!

The Absolute Beginners Inc. is again at your door! This time their demand is, however, simple. They have a C program, which computes the area of a circle (code shown below). They want it to be rewritten in Java.

```c
int
main(int argc, char **argv)
{
    int radius = 12.34;

    printf("Area of the circle with radius %f is: %f\n", radius, area(radius));

    return 0;
}

float
area(float r) {
    return 22 * r * r / 7;
}
```

Using Halstead's metrics estimate the effort required to recreate this program.

Learning Objectives:

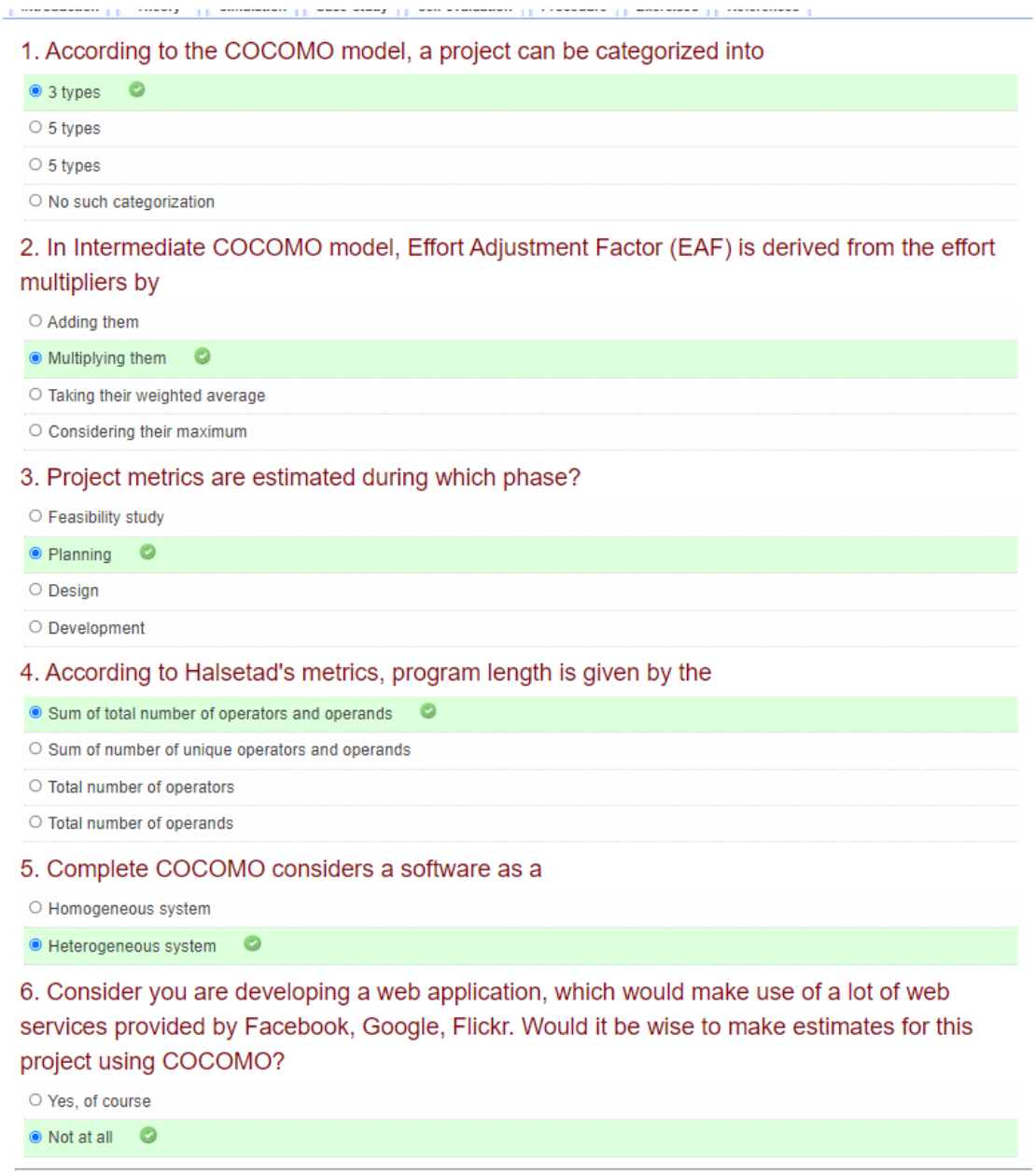
1. Determining estimated effort using Halstead's metrics

Submit

Parameter	Value
Total # of operators	30
Total # of operands	13
Total # of unique operators	14
Total # of unique operands	9
Program length	43
Program vocabulary	23
Volume	194.51
Difficulty	10.11
Effort	1966.5
Time to implement (in seconds)	109.25

Calculate

Self Evaluation :



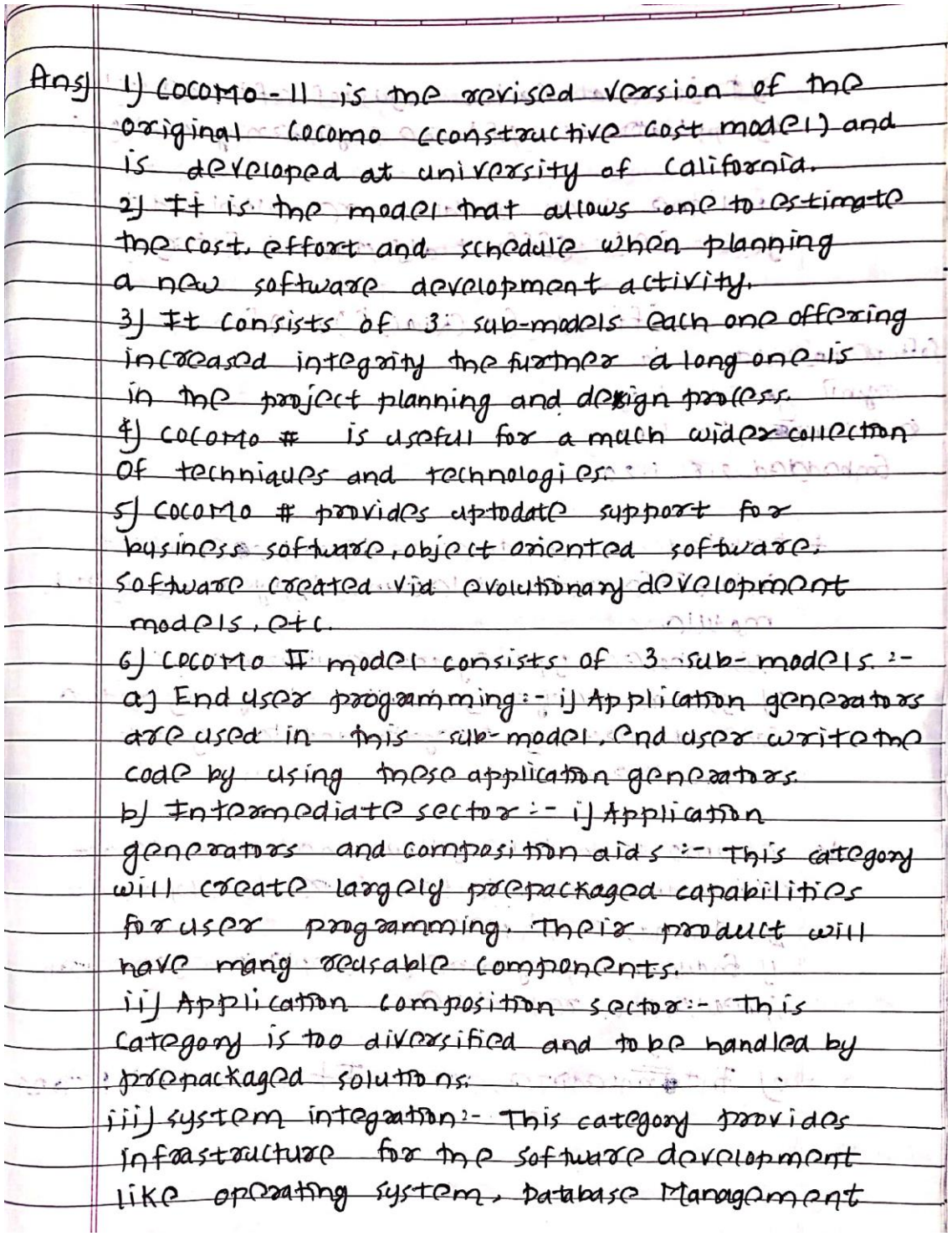
1. According to the COCOMO model, a project can be categorized into
   - (●) 3 types ✓
   - ( ) 5 types
   - ( ) 5 types
   - ( ) No such categorization
2. In Intermediate COCOMO model, Effort Adjustment Factor (EAF) is derived from the effort multipliers by
   - ( ) Adding them
   - (●) Multiplying them ✓
   - ( ) Taking their weighted average
   - ( ) Considering their maximum
3. Project metrics are estimated during which phase?
   - ( ) Feasibility study
   - (●) Planning ✓
   - ( ) Design
   - ( ) Development
4. According to Halsetad's metrics, program length is given by the
   - (●) Sum of total number of operators and operands ✓
   - ( ) Sum of number of unique operators and operands
   - ( ) Total number of operators
   - ( ) Total number of operands
5. Complete COCOMO considers a software as a
   - ( ) Homogeneous system
   - (●) Heterogeneous system ✓
6. Consider you are developing a web application, which would make use of a lot of web services provided by Facebook, Google, Flickr. Would it be wise to make estimates for this project using COCOMO?
   - ( ) Yes, of course
   - (●) Not at all ✓

Conclusion: Successfully understood the concept of how different estimations with respect to a project should be made. Also performed one of the estimation techniques with the help of online tool. Also gained some additional knowledge about the concept by performing the virtual lab.

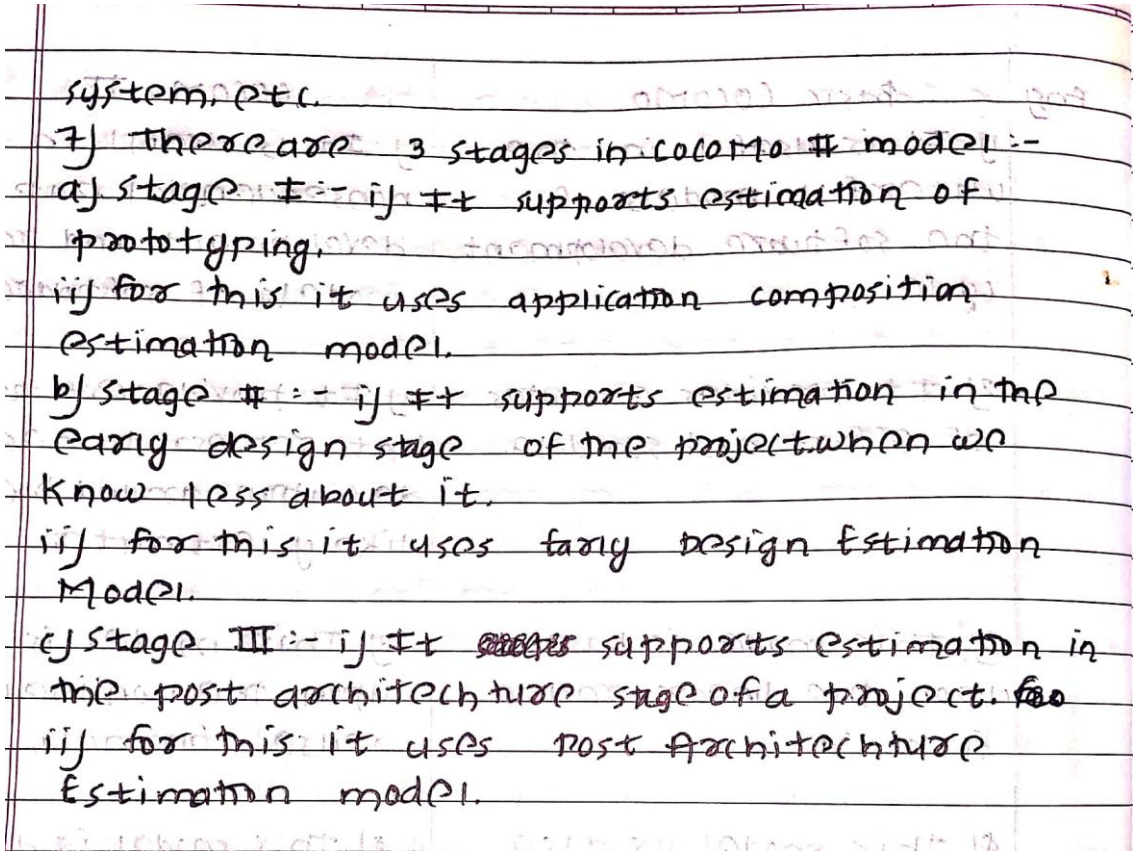
Post Lab Descriptive Questions

1. Explain COCOMO II model

ANS)



Ans) 1) Cocomo-II is the revised version of the original Cocomo (constructive cost model) and is developed at university of California.
2) It is the model that allows one to estimate the cost, effort and schedule when planning a new software development activity.
3) It consists of 3 sub-models each one offering increased integrity the further a long one is in the project planning and design process.
4) Cocomo # is useful for a much wider collection of techniques and technologies.
5) Cocomo # provides uptodate support for business software, object oriented software, software created via evolutionary development models, etc.
6) Cocomo II model consists of 3 sub-models :-
a) End user programming :- i) Application generators are used in this sub-model, end user write the code by using these application generators.
b) Intermediate sector :- i) Application generators and composition aids :- This category will create largely prepackaged capabilities for user programming. Their product will have many reusable components.
ii) Application composition sector :- This category is too diversified and to be handled by prepackaged solutions.
iii) system integration :- This category provides infrastructure for the software development like operating system, Database Management



system, etc.

7) There are 3 stages in COCOMO II model :-

a) Stage I :- it supports estimation of prototyping.

ii) for this it uses application composition estimation model.

b) Stage II :- it supports estimation in the early design stage of the project when we know less about it.

ii) for this it uses early design estimation Model.

c) Stage III :- it supports estimation in the post architecture stage of a project.

ii) for this it uses post architecture estimation model.

2. Compare the merits & limitations of basic COCOMO model & COCOMO II model.

ANS)

Ans) Basic Cocomo	Cocomo II
1) It is useful in the waterfall models of the software development cycle.	1) It is useful in nonsequential, rapid development and reuse models of software.
2) It provides estimates of effort and schedule.	2) It provides estimates that represent one standard deviation around the most likely estimate.
3) This model is based upon the linear reuse formula.	3) This model is based upon the nonlinear reuse formula.
4) This model is also based upon the assumption of reasonably stable requirements.	4) This model is also based upon reuse model which looks at effort needed to understand and estimate.
5) Effort equation's exponent is determined by 3 development modes.	5) Effort equation's exponent is determines by 5 scale factors.
6) Size of software stated in terms of lines of code.	6) Size of software stated in terms of object points, function points and lines of code.

3. Briefly explain the various types of efforts & cost estimation techniques used in Software Engineering

ANS)

Ans) Cost Estimation :- 1) It simply means a technique that is used to find out the cost estimates.
2) The cost estimate is the financial spend that is done on the efforts to develop and test software in software engineering.
3) cost estimation models are some mathematical algorithms or parametric equations that are used to estimate the cost of a product or a project.
4) Various techniques are :-
A) Empirical estimation technique :- a) Empirical estimation is a technique or model in which empirically derived formulas are used for predicting the data that are required and essential part of the software project planning step.
b) These techniques are usually based on the data that is collected previously from a project and also based on some guesses, prior experience with the development of similar types of projects and assumptions.
B) Heuristic technique :- a) Heuristic word is derived fr from a Greek word that means to discover. The heuristic technique is a technique or model that is used for solving problems, learning or discovery in the practical methods which are used for achieving immediate goals.

b) These techniques are flexible and simple for taking quick decisions through shortcuts and good enough calculations, most probably when working with complex data.

c) Analytical Estimation Technique:- a) Analytical estimation is a type of technique that is used to measure work. In this technique firstly the task is divided or broken down into its basic component operations or elements for analyzing.

b) Second, if the standard time is available from some other source, than these sources are applied to each element or component of work.

Effort estimation:- 1) Understanding the size and effort of a software project early on is a difficult problem. several different method exist, but no method is perfect.

2) Expert estimation:- a) Expert estimation means that an expert estimates how much effort a project requires.

b) The advantages of asking somebody else than the project manager to estimate a project is that some experts have deep knowledge about the problem at hand.

3) Top down estimation:- a) Top-down, analogy-driven estimation methods use experience from the past to make estimates for the future.

b) Analogy driven estimation methods require examples of completed IT projects to base the new estimates upon.

4) Bottom-up estimation :- a) Bottom-up Bottom-up estimation method take a project definition and examine what activities or deliverables need to be completed in order to achieve the project's objective.
b) One keeps breaking up the project activities or deliverables into smaller subactivities or partial deliverables until each sub-activity of partial deliverables requires less than 2 weeks of effort.
5) Parametric estimation methods :- a) Parametric estimation methods use a model or algorithm that takes as input some aspects of the project (such as the required functionality and the quality that is expected).
b) The model (a formula) or algorithm (computational steps) then produce an estimate based on those inputs alone.