- Assignments must be completed individually.
- No late assignments will be accepted.
- Provide **concise** answers to the following questions. Use **point form** whenever possible.
- Submit your completed solutions to **Crowdmark**.

1. Give one example of each of the following impediments to portability:

[1] (a) a hardware incompatibility

[1] (b) an operating system incompatibility

[1] (c) a numerical system incompatibility

[1] (d) a compiler incompatibility

[2] 2. Briefly explain why developing your software product using the UNIX operating system maximizes the probability that your software product will be **portable** when it is complete.

[3] 3. Your manager tells you that a client has requested a quotation for developing a software product **before the analysis workflow has been completed**. Give some points to persuade your manager that doing this is a bad idea. You may find §9.1 in the text, and Lecture 21 in the Lecture Notes, to be helpful here. You do **not** need to cite specific statistical data in your answer.

[4] 4. Give reasons why **lines of code (LOC)** is not reliable as the **sole** measure of the size of a software product.

5. This question is about measuring the size of a software product using **function points**, as in §9.2.1 of the text and Lecture 21 of the Lecture Notes.

[4] (a) Compute the **unadjusted function points (UFP)** for a software product having the following function point counts in conjunction with Figure 9.3 in the text (reproduced here). Show your work.

Figure 9.3 - Table of Function Point Values

	Level	of	Complexity
Component	Simple	Average	Complex
Input item	3	4	6
Output item	4	5	7
Inquiry	3	4	6
Master file	7	10	15
Interface	5	7	10

Function Point Counts to Use

	Level	of	Complexity
Component	Simple	Average	Complex
Input item	4	5	2
Output item	8	6	2
Inquiry	9	7	1
Master file	3	2	1
Interface	3	4	3

[2] (b) Compute the **technical complexity factor (TCF)** using the given counts for each factor in Figure 9.4 from the text (reproduced here). Show your work.

Figure 9.4 (augmented) - Technical factors for function point computation

Factor	Name	Count to Use
1	Data communication	3
2	Distributed data processing	2
3	Performance criteria	1
4	Heavily utilized hardware	2
5	High transaction rates	2
6	Online data entry	3
7	End-user efficiency	2
8	Online updating	2
9	Complex computations	2
10	Reusability	1
11	Ease of installation	2
12	Ease of operation	2
13	Portability	2
14	Maintainability	2

[2] (c) Use the results of parts 5a and 5b to compute the **function points (FP)** for the given software product. **Round your answer up** to the next whole number of **function points**. Show your work.

6. This question is about estimating the effort to develop a software product using **Intermediate COCOMO**, as in §9.2.3 of the text and Lecture 22 of the Lecture Notes.

[2] (a) Compute the **nominal effort** for a software product having

- **organic** development mode (with multiplier 3.3),
- **exponent** 1.05, and
- 9500 LOC (i.e. 9.5 KDSI).

Round your answer up to the next whole number of **person-months**. Show your work.

[2] (b) Use part a) to compute the **estimated effort** using the given multipliers for each factor in Figure 9.6 from the text (reproduced here).

Figure 9.6 (augmented) - Intermediate COCOMO software development effort multipliers

			Rating				
Cost	Very				Very	Extra	Rating
Drivers	Low	Low	Nominal	High	High	High	to Use
Product Attributes							
-Required software reliability	0.75	0.88	1.00	1.15	1.40		Nominal
-Database size		0.94	1.00	1.08	1.16		High
-Product complexity	0.70	0.85	1.00	1.15	1.30	1.65	Low
Computer Attributes							
-Execution time constraint			1.00	1.11	1.30	1.66	High
-Main storage constraint			1.00	1.06	1.21	1.56	Nominal
-Virtual machine volatility		0.87	1.00	1.15	1.30		Nominal
-Computer turnaround time		0.87	1.00	1.07	1.15		Low
Personnel Attributes							
-Analyst capabilities	1.46	1.19	1.00	0.86	0.71		Low
-Applications experience	1.29	1.13	1.00	0.91	0.82		High
-Programmer capability	1.42	1.17	1.00	0.86	0.70		Nominal
-Virtual machine experience	1.21	1.10	1.00	0.90			Low
-Programming language experience	1.14	1.07	1.00	0.95			High
Project Attributes							
-Use of modern programming practices	1.24	1.10	1.00	0.91	0.82		Very High
-Use of software tools	1.24	1.10	1.00	0.91	0.83		High
-Required development schedule	1.23	1.08	1.00	1.04	1.10		High

Round your answer up to the next whole number of **person-months**. Show your work.

[2] 7. When I worked at SunLife Financial, all teams within IT adhered to strict standards with respect to how all source code was commented. Give one reason which would have motivated the company to adopt this policy.

[2] 8. You are going to announce that you have invented a new software development paradigm called **TomorrowForToday**, and that it has solved all the problems of the current Object-Oriented Paradigm. In your promotional material for the new paradigm, you will state four advantages of your TomorrowForToday paradigm. Which two advantages would you include in your promotional material? Note, you only need to state what the advantages are; you do **not** need to give specific details on how these advantages are achieved.

[4] 9. Imagine that it is 1970 and that you are managing a software development team. To date, your team has used the **code-and-fix** life-cycle development model which was prevalent at the time. State reasons which you would give to your manager to justify why it would benefit your team and its clients to adopt the newly-introduced **waterfall** life-cycle development model.