Question 1 [50 marks in total]

A store in city offers different discounts depending on the purchases made by the individual. In order to test the software program that calculates the discounts, it is possible to identify the “price range” of purchase values that earn the different discounts. For example, if a purchase is in the price range of €1 up to and including €50 it has no discounts, a purchase in the price range of over €50 and up to €200 has a 5% discount, and purchases of price range €201 and up to €500 have a 10% discounts, and purchases of price range €501 to €2500 have 15% discounts. No items are sold outside these price ranges 这些价格范围以外没有出售记录 and if an incorrect price is entered the software returns a discount value of -1% to indicate that an error case has occured.

Considering the Black-Box Software testing method of Equivalence Partitions 等价类划分 for this program write the input and output partitions. Then write out suitable tests ensuring to outline the partitions covered with each set of test data and highlight any error case with an asterisk. [15 marks] **[1 mark for each partition and 1 mark for each test data. There could be two extra marks but the maximum mark is 15 for this question] Note: the numerical values for the test data should be anywhere within each partition**

Test Cases	**Input Partitions**	**Range**
	Parameter - Price range	
EP1*		Integer.MIN_VALUE…0
EP2		1…50
EP3		51…200
EP4		201…500
EP5		501…2500
EP6*		2501…Integer. MAX_VALUE
	Output Partitions	
	Parameter - Discount	
EP7*		-1
EP 8		0
EP 9		5
EP 10		10
EP 11		15

Test No.	**Test Cases Covered**	**Inputs**	**Expected Outputs**
		Price range	*Discount*
1	EP2, 8	25	0
2	EP3, 9	125	5
3	EP4 10	350	10
4	EP5, 11	1250	15
5	EP1*, 7*	-50	-1
6	EP6*, 7*	3000	-1

(b) A program CheckEquilateral takes three integers as input that represents the lengths of the three sides of a triangle. The program first determines if the three numbers are valid inputs, that is, the length of any sides must not be 0 or a negative number. If not, it will return 'Not a triangle'. Otherwise, it will check whether it is an "Equilateral triangle" (three sides are the same length). Using the Causes and Effects as given below, complete the Truth Table underneath for the black box technique of Combinational Testing (note: impossible test cases have been eliminated). Following this, create suitable Test Data. [10 marks]

Causes	**Effects**
Side1>0	Not a triangle
Side2>0	Equilateral
Side3>0	Not Equilateral
Side1=Side2	
Side2=Side3	
Side1=Side3	

	Rules										
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Causes											
Side1>0	F	*	*	T	T	T	T	T	T	T	T
Side2>0	*	F	*	T	T	T	T	T	T	T	T
Side3>0	*	*	F	T	T	T	T	T	T	T	T
Side1=Side2	*	*	*	F	F	F	F	T	T	T	T
Side2=Side3	*	*	*	F	F	T	T	F	F	T	T
Side1=Side3	*	*	*	F	T	F	T	F	T	F	T
Effects											
Not a Triangle	T	T	T	F	T	T	-	F	-	-	F
Equilateral	F	F	F	F	F	F	-	F	-	-	T
Not Equilateral	F	F	F	T	T	T	-	T	-	-	F

The shaded Rules indicate impossible test cases. These are removed below.

	Rules							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Causes								
Side1>0	F	*	*	T	T	T	T	T
Side2>0	*	F	*	T	T	T	T	T
Side3>0	*	*	F	T	T	T	T	T
Side1=Side2	*	*	*	F	F	F	T	T
Side2=Side3	*	*	*	F	F	T	F	T
Side1=Side3	*	*	*	F	T	F	F	T
Effects								

Not a Triangle	T	T	T	F	T	T	F	F
Equilateral	F	F	F	F	F	F	F	T
Not Equilateral	F	F	F	T	T	T	T	F

Each Rule is a Test Case.

[1 mark for line in the truth table for Effects and 1 mark for each line of test data. There could be one extra mark but the maximum mark is 10 for this question]

Test Data

Test No.	**Test Cases Covered**	**Inputs**			**Expected Outputs**
		Side1	*Side2*	*Side3*	*Triangle Type*
1	TT4	5	4	3	Not Equilateral
2	TT5	2	1	2	Not Equilateral
3	TT6	1	2	2	Not Equilateral
4	TT7	2	2	1	Not Equilateral
5	TT8	1	1	1	Equilateral
6	TT1	-1	1	1	Not a Triangle
7	TT2	1	-1	1	Not a Triangle
8	TT3	1	1	-1	Not a Triangle

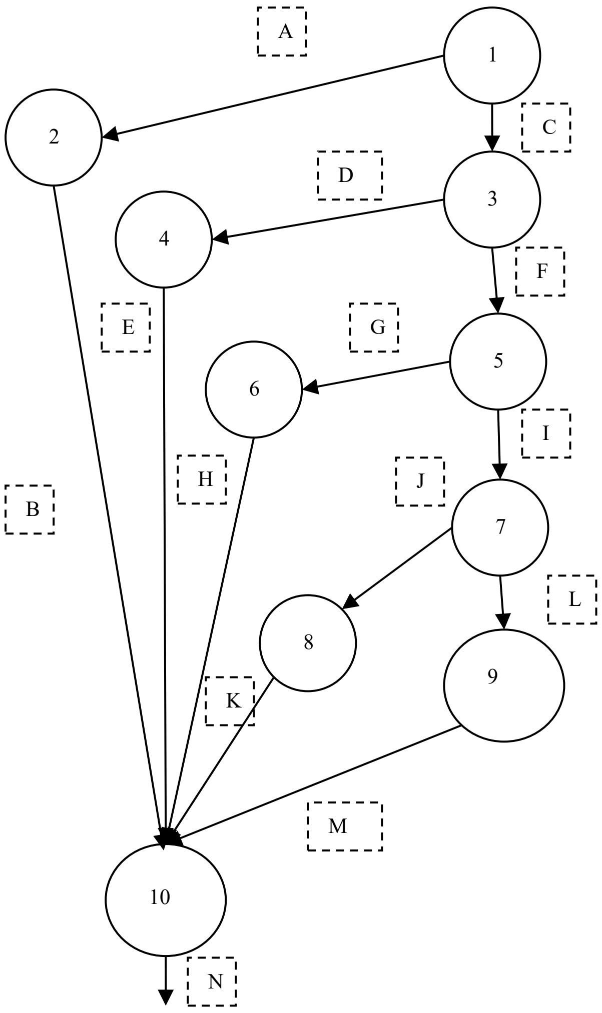
(c)The software program Grade combines an exam and coursework mark into a single grade. The values for exam and coursework are integers. If the exam or coursework mark is less than 50% then the grade returned is a ‘Fail’. To pass the course with a ‘Pass, C’, the student must score from 50% to 60% in the exam, and at least 50% in the coursework. They will pass the course with ‘Pass, B’, if they score more than 60% in the exam and 50% or more in the coursework. In addition to this 除此之外, if the score of both the exam and the coursework is 90% or greater they are awarded a ‘Pass, A’. Input values that are less than 0 or greater than 100 for either the exam or coursework are invalid and the program will return a message to say ‘Marks out of range’.

Draw a Control Flow Graph (CFG) for the program Grade() using the source code given below. [10 marks] Then, derive 生成 test cases and test data for the following White Box Software testing methods: [15 marks]

i. Statement Testing

Line No.	**Code**
1)	`public static String Grade (int exam, int course) {`
2)	`String result="null";`
3)	`if ( (exam<0) || (exam>100) || (course<0) || (course>100) )`
4)	`result="Marks out of range";`
5)	`else {`
6)	`if ( (exam<50) || (course<50))`
7)	`result="Fail";`
8)	`else if (exam <= 60)`
9)	`result="Pass,C";`
10)	`else if ((exam>=90) && (course>=90))`
11)	`result="Pass,A";`
12)	`else`
13)	`result="Pass,B";`
14)	`}`
15)	`return result;`
16)	`}`

The CFG is [10 marks]



CFG

Grade

Node	Lines
1	1…3
2	4
3	5…6
4	7
5	8
6	9
7	10
8	11
9	12...14
10	15...16

Each node in the CFG is a test case. The tests cases and test data are **[15 marks/ 3 marks for each set of cases covered and test data values. Test data numerical values are arbitrary** 任意 **and are picked so that specific text cases are executed]**

Test No.	**Test Cases Covered**	**Inputs**		**Expected Outputs**
		exam	*course*	*Result*
1	SC1, 2, 10	-1	50	Marks out of Range
2	SC[1], 3, 4, [10]	40	50	Fail
3	SC[1], [3], 5, 6, [10]	55	50	Pass, C
4	SC[1], [3], [5], 7, 8, [10]	95	95	Pass, A
5	SC[1], [3], [5], [7], 9, [10]	80	50	Pass, B

Only do two Questions from the following:

Question 2

(a) Say why Exhaustive 穷尽 software testing is generally very difficult. Use an example to illustrate your answer. [7 marks]

This is generally not feasible 可行的 as it would take too long or require too much memory space. **[2 marks]**

Consider a method bound() as defined below: **[5 marks for an example similar to below]**

```
// return a value of x bounded by the upper and lower values
// return lower if x<=lower
// return upper if x>=upper
// return x if lower<x<upper
long bound(int lower, int x, int upper);
```

If int is defined as a 32-bit, signed integer, then each input parameter has 2^{32} possible input values. Exhaustive testing would require using every possible combination of input values, leading to 2^{96} tests. Consider the situation that, on average, each test took 1 nanosecond to execute – possible on a modern, multi-core processor – then the test would take 2^{96} (or about 10^{29}) nanoseconds to complete. This is approximately 10^{12} years. The sun is only $4.6 * 10^9$ years old. So clearly this is not feasible, even for a simple function.

(b) Describe two significant 显著 differences between Black-box testing and White-box testing. [8 marks]
Take two items from the left column **[2 marks each]** and two items from the right column **[2 marks each]**

Black Box Testing	**White Box Testing**
Tests are only dependent on the specification	Tests are dependent on both the implementation and the specification.
Tests can be re-used if the code is updated to x a fault, or additional functionality is added.	Tests are generally invalidated by any changes to the code.
Tests only require the specification to be developed.	Tests require that the code is written prior to developing the tests.
Tests do not ensure that all the code components have been exercised However, and thus are liable to missing \errors of commission" where extra functionality has been added to the code that is not in the specification.	Tests ensure that all the (selected) components have been exercised, However they do not ensure that the specification has been fully covered. They are thus liable to missing \errors of omission" where required functionality has not been implemented
In general, it is dicult to automate measurement of the effectiveness of black-box tests (i.e. how much of the specication has been covered).	In general, it is relatively easy to automate measurement of the effectiveness of white-box tests (i.e. how much of the implementation has been covered).

(c) A useful Software testing technique that can be used to check the value or strength of a set of software test data is "Fault Insertion"故障注入 or "Mutation testing"突变测试. Offut suggested five categories of mutants that are sufficient for this type of testing. What are they? [10 marks] **[10 marks total/2 marks for each of the categories]**

1. Absolute value
   Use the Absolute value or a variant ABS(), -1*ABS(), 0
2. AOR (arithmetic operator replacement)算术运算符替换
   Replace one of the arithmetic operators by one of the others: + - * / %
3. LCR (logical connector replacement)逻辑运算符替换
   Replace one of the logical operators by one of the others: (a && b) or (a || b) becomes (a other-op b), (a), (b), (a op true), (a op false), (true op b), (false op b)
4. ROR (Relational operator replacement)关系运算符替换
   Replace one of the relational operators by one of the others: (a compare b) becomes a [< <= > >= != ==] b, true, false
5. UOI (unary operator insertion)
   Insert a unary operator before the expression: insert "! - ++ --" before expression

Question 3

(a) Explain the approach to software development that is known as ‘Big Bang’. Give two drawbacks 劣势 to using it. [7 marks]

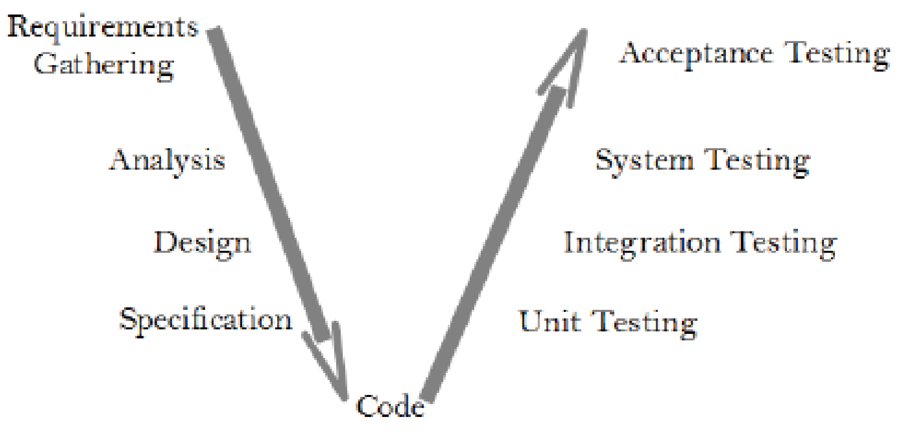
Big Bang definition [3 marks]
Big Bang – this means waiting until all the code has been written and then to test the finished product all at once 同时. It is an attractive 吸引人的 option to developers because testing activities do not hold back the progress towards completing the product.

Drawbacks [4 marks total/2 marks each]
The likelihood that the product will work, or even be close to working can be very low because no intermediate 中间 checks have been made. This is particularly dependent on program complexity and program size.

Additionally, if tests do reveal faults in the program, it is much more difficult to identify their source. It is then necessary to search through the complete program to locate them.

(b) The V-model of software development is similar to the Waterfall model but it is more test-oriented emphasizing verification and validation throughout the process. Give an outline of the V-model in the form of a diagram. There are four documents produced, each of which is associated with pairs of phases in the model. Name these documents and say how they are used in implementing the model. Give one advantage and one disadvantage of the model. [18 marks]

An outline of the v-model is given in the diagram **[2 marks]**



Four documents are produced that can be associated with pairs of phases in the model. **[12 marks in total/ 3marks for each]**

Requirements Gathering produces the User Requirements Specification (URS 用户需求规格书), which is both the input to Analysis, and the basis for Acceptance Testing.

Analysis produces the System Specification (SS) – also know as the Software Requirements Specification (SRS 软件需求规格书) – which is both the input for Software Design, and the basis for System Testing.

Design produces the System Design Specification (SDS), which is both the input for the detailed Specification phase and the basis for Integration Testing.

The Specification activity produces the Detailed Design Specifications (DDS), which are both used to write the code, and also are the basis for Unit Testing.

One advantage from these [2 marks]
It is simple and easy to manage due to the rigidity of the model,

It encourages 促进 Verification and Validation at all phases:

Each phase has specific deliverables and a review process.

It gives equal weight to testing alongside development rather than treating it as an afterthought.

One disadvantage from these [2 marks]
There is no working software produced until late during the life cycle

It is unsuitable where the requirements are at a moderate to high risk of changing.

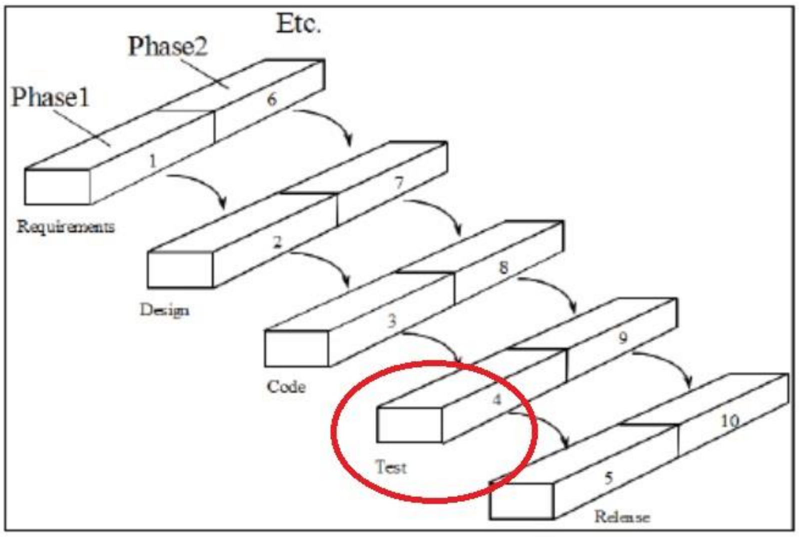
It is a poor model for long, complex and object-oriented projects

Question 4

(a) How does the Incremental development model incorporate Software testing? Use a diagram to illustrate your answer. Give one advantage and one disadvantage of this approach to Software development. [10 marks]

[6 marks total/2 marks each + 2 marks for the diagram]
Testing is an important part of the incremental model and is carried out at the end of each iteration 迭代. This means that testing begins earlier in the development process and that there is more of it overall.

Much of the testing is of the form of regression testing 回归测试, and much re-use can be made of test cases and test data from earlier increments.



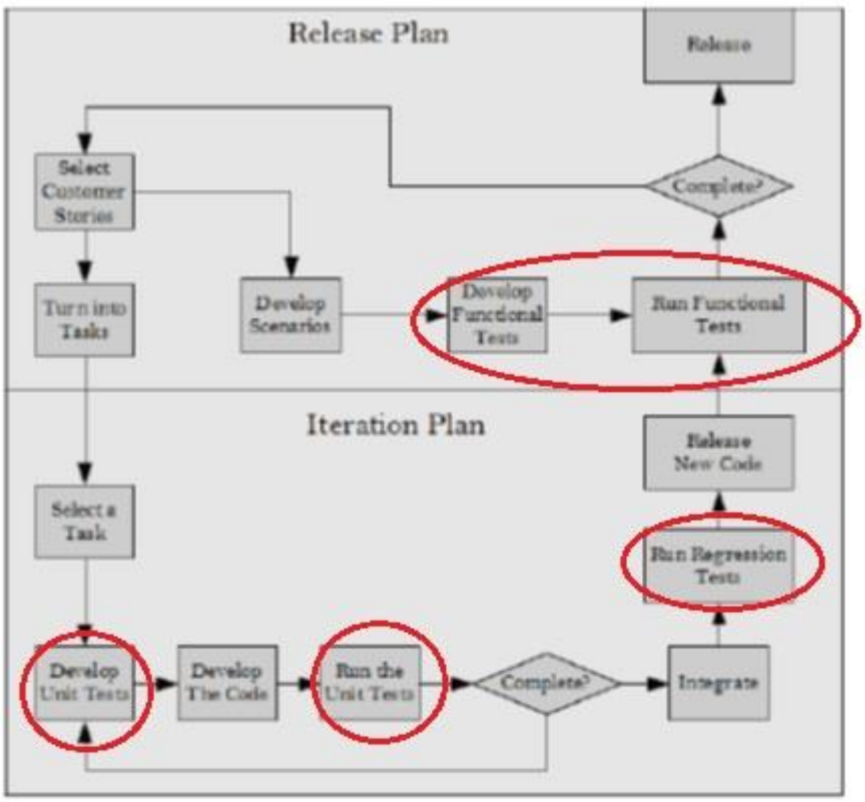
Either of the following is an advantage [2 marks]
A major advantage of the incremental model is that the product is written and tested in smaller pieces, reducing risk and allowing for change to be included easily

The customer or users is involved from the beginning which means the system is more likely to meet their requirements and they themselves are more committed to the system

Either of the following is a disadvantage [2 marks]
It can be difficult to manage because of the lack of documentation in comparison to other models

The continual change to the software can make it difficult to maintain as it grows in size.

(b) Draw a diagram of the XP, Extreme Programming 极 限 编 程 , software development process. Ensure to highlight in your diagram the points at which software testing is carried out. [9 marks] **[6 marks for the diagram and 3 marks for highlighting the points of testing]**



(c) Explain <u>one</u> benefit and <u>one</u> disadvantage to the DevOps technique. [6 marks]

Any one of these is a benefit [3 marks]
Faster Time to market – reduced cycle times and higher deployment rates

Increased Quality 提高质量 – better availability of the product as it is being created, increased change success rate and fewer failures

Increased organizational effectiveness 提高组织的有效性 – more time spent on value adding activities and greater value being delivered to the customer

Any one of these is a Drawback [3 marks]
DevOps requires high level of coordination between various functions of the deliverable chain.

A need to master the various automation and continuous integration tools 各种自动化和连续集成工具