Software Engineering

Books or notes are **not** allowed.
Write only on these sheets. **Concise** and **readable** answers please.

Surname, name, matricola ____

1 Non Functional requirement
A mobile application is designed to remotely control an antiTheft system in a house, that controls if someone enters the house and, in case, triggers an alarm. The main functions are to monitor the state of the house (doors and windows are open or closed), switch the antiTheft system on or off, monitor the state of the alarms, switch the alarm off.

Define the most important (max 3) non functional requirements for the application.

Security: access to functions should be possible only for selected users (ex: owner of house, maintenance personnel, police)

Usability: non specialist user (ex owner of house) should be able to use the application with <5 minutes training

Reliability: <1 failure per calendar year per user

Also important NFR: portability (available on Android and IoS) , robustness (

2 Black box

Define black box tests for the following function, using equivalence classes and boundary conditions.

int computeTax (int totalIncome, int deductible, int taxDeductible)

The function computes the amount of money to be paid as taxes by a person, in euro. All values are integers, cents are rounded down.
totalIncome is the total amount of money gained by a person in a fiscal year. The tax is computed as 30% of totalIncome – deductible, however the first 10.000 euro are tax free.

Ex computeTax (10000, 0,0) → 0
computeTax(20000, 0,0) → 3.000 (30% of 20.000- 10.000)

deductible is the amount of expenses that the person declares. The deductible is subtracted by totalIncome, however the maximum deductible per year is 4000. In any case the tax to be paid cannot be negative

computeTax(20000, 2000, 0) → 2.400 (30% of (20.000-2000) - 10.000)
computeTax(20000, 5000, 0) → 1.800 (30% of (20.000-4000) - 10.000)

taxDeductible is another category of expenses that the person declares. These expenses are treated in a different way. 20% of these expenses is subtracted to the amount of taxes due. However this applies only to a max of 2000 taxDeductible per year

computeTax(20000, 0, 1000) → 2.800 = 3000 (30% of (20.000-0) - 10.000) - 200 (20% of 1.000)
computeTax(20000, 0, 3000) → 2.600 = 3000 (30% of (20.000-0) - 10.000) - 400 (20% of 2000)

total income [minInt, 0[, [0, 10.000], [10.000, maxInt]
deductible [minInt, 0[, [0, 4.000],] 4.000, maxInt]
taxDeductible [minInt, 0[, [0, 2.000],]2.000, maxInt]

Total income	deductible	taxDeductible	**0,3(totalIncome-10000-deductible)-0,2*taxDeductible>=0**	V / I	Test cases
[minInt, 0[	[minInt, 0[	[minInt, 0[	-	I	(-1, -1. -1) error
[0, 10.000]	[0, 4.000]	[0, 2.000]	T	V	(1, 4000, 0), 0 (1, 0, 2000), 0 (1, 4000, 2000), 0
			F	V	(1000, 2000, 1000),
		]2.000, maxInt]	T	V	(1,0, 4000) , 0 (1, 4000, 4000), 0
			F	V	(1000, 2000, 4000),
	]4.000, maxInt]	[0, 2.000]	T	V	
			F	V	
		]2.000, maxInt]	T	V	
			F	V	
]10.000, maxInt]	[0, 4.000]	[0, 2.000]	T	V	

			F	V	
		]2.000, maxInt]	T	V	
			F	V	
	]4.000, maxInt]	[0, 2.000]	T	V	
			F	V	
		]2.000, maxInt]	T	V	
			F	V	

Remark that test cases for Invalid partitions MUST be written and applied

3-White box

For the following function define the control flow graph, and define test cases to obtain the highest possible node coverage, edge coverage, multiple condition coverage, loop coverage, path coverage.
For the test cases, **write only the input value**.

```
1      int foo(int i, int j) {
2          int var = 'A';
3          if (i <0 || j < 0) return -1;
4          for ( ; i >= 2; i--) {
5              for (j = 0; j < i; j++){
6                  printf("%c ", (var + j));
7              }
8              printf("\n");
9          }
10         return 0;
11     }
```

Coverage type	Number of test cases needed to obtain 100% coverage	Coverage obtained with test cases defined (%)	Test cases defined

Node	2	100	T1(2,1)
			T2(-1, 1)

Edge	2	100	T1, T2
Multiple condition line 3	4	100	T1 FF T2 TF T3(1, -1) FT T4(-1, -1) TT
Loop line 4	3	100	Try no enter T5(1, 2) Enter 1 T1 Enter many T6(4, 2)
Loop line 5	3	33%	Try no enter – not feasible Enter 1 - not feasible Enter many T6(4, 2)
Path	Depends on I and j, each pair (with I >2) defines a new path. In short the number of paths are in the order of maxint*maxint	Close to zero with test cases above	

4 What is an ‘oracle’ ? How can it be implemented?

An oracle is a (theoretical) entity capable of providing the correct result of a function, given the inputs. The practical implementation of an oracle is always subject to error. Practical implementations are humans, or other software applications (ex previous versions or legacy version of the same function, or mathematical formulas implemented by other similar programs)

5 Considering a software project, what is the relationship between effort and size?

Effort = f(size)   where f is linear or more than linear
Productivity as a fixed number implies a linear relationship between effort and size, but this is not always true

6 Describe the 'pipe and filter' architectural pattern

See slides

7 What are the main differences between a waterfall process and an agile process?

Waterfall:
- only sequential activities (no parallelism)
- one big iteration (starting with the requirements and ending with a big integration + delivery)
- big importance to documentation and fixed price business interaction

- suitable for large size projects

Agile:

- parallel activities
- many iterations (of 24 hours and each iteration is within a sprint of 3-4 weeks ending with a delivery)
- not big importance to documentation (time and material busiess interaction)
- suitable for small projects

See slides

8 The estimated effort for a software project is 1000 person hours.   On the project can work 2 people full time, 2 people part time (50%). Make an estimate of calendar duration of the project, and explain it   Assume 8 working hours per day, 5 days per week.

2 people full time + 2 people half time = 3 FTE
One calendar week = 8x5 = 40 ph
Available ph per week = 3* 40 = 120ph

1000 / 40*3 = 8.3 calendar weeks = approx 2 calendar months
In practice the duration of the project can be estimated to no less than 2 months, likely at least 3 months considering the staffing profile (not all personnel can work in parallel during the whole project)