1 Specificați si testați funcția: (**1.5p**)

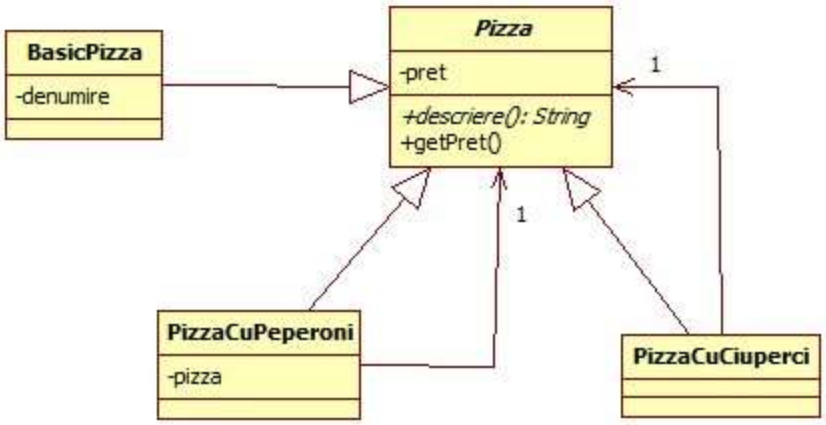
```
vector<int> f(int a) {
	if (a < 0)
		throw std::exception("Illegal argument");
	vector<int> rez;
	for (int i = 1; i <= a; i++) {
		if (a % i == 0) {
			rez.push_back(i);
		}
	}
	return rez;
}
```

2 Indicați rezultatul execuției pentru următoarele programe c++. Daca sunt erori indicați locul unde apare eroarea si motivul.

```
//2 a  (1p)
#include <iostream>
using namespace std;
int except(int v) {
	if (v < 0) {throw 1;  }
	else if (v > 0){
	   throw std::exception ("A");
	 }
	return 0;
}
int main(){
	try {
	   cout << except(1 < 1);
	   cout << except(-5);
	   cout << except(5);
	}catch (std::exception& e) {
		cout << "A";
	}catch (int x) {
		cout << "B";
	}
	cout << "C";
    return 0;
}
```

```
//2 b  (0.5p)
#include <iostream>
using namespace std;
class A {
	int x;
public:
	A(int x) : x{ x } {}
	 void print(){cout<< x <<",";}
};
A f(A a) {
	a.print();
	a = A{ 7 };
	a.print();
	return a;
}
int main() {
	A a{ 5 };
	a.print();
	f(a);
	a.print();
}
```

3 Se da diagrama de clase UML: (**4p**)



- Clasa abstracta **Pizza** are o metoda pur virtuala descriere()
- **PizzaCuPeperoni** si **PizzaCuCiuperci** conțin o pizza si metoda descriere() adaugă textul "cu peperoni" respectiv "cu ciuperci" la descrierea pizzei conținute. Prețul unei pizza care conține peperoni creste cu 2 Ron, cel cu ciuperci costa in plus 3 RON.
- Clasa **BasicPizza** reprezintă o pizza fără ciuperci si fără peperoni, metoda descriere() returnează denumirea pizzei. În pizzerie exista 2 feluri de pizza de baza: Salami si Diavola, la prețul de 15 respectiv 20 RON.

Se cere:

1. Scrieți codul C++ **doar pentru clasele Pizza si PizzaCuPeperoni. (0.75p)**
2. Scrieți o **funcție** C++ care creează o comanda (**returnează o lista de pizza**) care conține: o pizza „Salami" cu ciuperci, o pizza „Salami" simplă, o pizza „Diavola" cu peperoni si ciuperci. **(0.5p)**
3. Scrieți programul principal care creează o comandă (folosind funcția descrisa mai sus), apoi tipărește descrierea si prețul pentru fiecare pizza in ordinea descrescătoare a preturilor. **(0.25p)**

Obs. Creați doar metode si atribute care rezultă din diagrama UML (adăugați doar lucruri specifice C++ ex: constructori). Nu adăugați câmpuri, metode, nu schimbați vizibilitatea, nu folosiți friend. Folosiți STL unde exista posibilitatea.
Detalii barem: **1.5p** Polimorfism, **1p** Gestiunea memoriei, **1.5p** Restul

4 Definiți clasa Catalog astfel încât următoarea secvență C++ sa fie corectă sintactic si să efectueze ceea ce indica comentariile. (**2p**)

```
void catalog() {
      Catalog<int> cat{ "OOP" };//creaza catalog cu note intregi
      cat + 10; //adauga o nota in catalog
      cat = cat + 8 + 6;
      int sum = 0;
      for (auto n : cat) { sum += n; } //itereaza notele din catalog
      std::cout << "Suma note:" << sum << "\n";
}
```