1. **Cand decidem sa utilizăm constructori de copiere intr-o clasa? Dați un mic exemplu;**

   Folosim constructorul de copiere atunci când avem nevoie să creăm un nou obiect care să fie o copie exactă a unui alt obiect deja existent.

   ## Cazuri în care este necesar un constructor de copiere:

- **Gestionarea memoriei dinamice** (evităm **shallow copy** → copiere superficială).
- **Transmiterea obiectelor prin valoare** în funcții.
- **Returnarea unui obiect dintr-o funcție**.
- **Copierea obiectelor care conțin pointeri** (pentru a evita problemele legate de alocarea memoriei).

```
class Persoana {
private:
    string nume;
    int varsta;
```

```
   Persoana(const Persoana& p) {
       nume = p.nume;
       varsta = p.varsta;
       cout << "Constructor de copiere apelat!" << endl;
   }
```

2. **Ce operatori noi sunt introdusi in standardul C++, fata de cei existenti in C? Explicatii succinte.**

Operator	Descriere	Exemplu
`::`	Rezoluția de domeniu (acces la membrii unei clase sau namespace)	`std::cout << "Hello";`
`new / delete`	Alocare și dealocare dinamică de memorie	`int* p = new int(10); delete p;`
`operator()`	Functori (obiecte apelabile ca funcții)	`obiect();`
`operator[]`	Indexare personalizată pentru clase	`obiect[2] = 5;`
`->* / .*`	Pointer la membru de clasă	`p->*ptrFunc;`
`dynamic_cast, static_cast, reinterpret_cast, const_cast`	Conversii de tip avansate	`Base* b = dynamic_cast<Base*>(ptr);`

3. **Supradefiniti operatorul de indexare pentru o posibila clasa Test; Declarati clasa cu o structura minimala și definiti doar operatorul în cauză;**

```
class Test {
private:
    int valori[5];
```

```
    int& operator[](int index) {
        if (index < 0 || index >= 5) {
            cout << "Index invalid!\n";
            exit(1);
        }
        return valori[index];
    }
};
```

4. **Dati un exemplu de o clasa cu membri statici. Cum pot fi folosiți acești membri?**

```
class ContBancar {
private:
    static int numarConturi;  // Membru static

public:
    ContBancar() {
        numarConturi++;  // Crește la fiecare obiect creat
    }

    static int getNumarConturi() {  // Metodă statică
        return numarConturi;
    }
};

int ContBancar::numarConturi = 0;  // Inițializare statică

int main() {
    ContBancar c1, c2, c3;
    cout << "Număr conturi: " << ContBancar::getNumarConturi() << endl;
    // Output: 3
}
```

5. **Ce reprezintă tipul referinta? Dar tipul pointer?**

Concept	Explicație	Exemplu
Referință (`&`)	Un alias pentru o variabilă existentă	`int x = 10; int &ref = x;`
Pointer (`*`)	O variabilă care stochează adresa altui obiect	`int* p = &x;`

6. **Utilitatea operatorului de copiere. Un scurt exemplu.**

```
class Exemplu {
    int x;
public:
    Exemplu(int val) { x = val; }  // Constructor normal

    Exemplu(const Exemplu& other) {  // Constructor de copiere
        x = other.x;
    }
```

7. **Cum este utilizat sistemul de conversie a tipurilor?**

- **`static_cast<T>(x)` → Conversii simple și sigure.**
- **`dynamic_cast<T>(x)` → Conversii între tipuri legate prin moștenire.**
- **`reinterpret_cast<T>(x)` → Conversii brute (periculoase).**
- **`const_cast<T>(x)` → Adaugă sau elimină `const` dintr-un obiect.**

```
double x = 10.5;
int y = static_cast<int>(x); // Conversie sigură de la double la int
```

8. **Dati un exemplu de o clasa ce implementeaza conceptul de numar complex;**

```
class Complex {
private:
    double re, im;

public:
    Complex(double r = 0, double i = 0) : re(r), im(i) {}

    void afisare() { cout << re << " + " << im << "i\n"; }
};
```

9. **Pentru urmatoarele concepte stabiliti ierarhia (structura) de clase:**

- **O aplicatie ce lucreaza cu trei concepte: Persoana, Profesor, Student si**
- **O alta aplicatie ce lucreaza cu conceptul de Student ce are note la 5 Discipline**

În această aplicație, avem o clasă de bază numită Persoana, iar celelalte clase, Profesor și Student, vor fi clase derivatedin Persoana.

```
      Persoana
      /      \
Profesor    Student
```

În acest caz, vom crea o clasă Student, care va conține un vector sau un tablou pentru notele la cele 5 discipline.

```
       Student
         |
    NoteDiscipline
```