Problema 5:

Scrieți codul necesar pentru ca următoarea funcție **main()**

```
int main() {
    SimpleList<int> l = { 1,3,5,7,9 };
    for (auto i : l) cout << i << " "; cout << endl;
    ((((l += 2) += 4) += 6) += 8);
    for (auto i : l) cout << i << " "; cout << endl;
    l += {1, 2, 3};
    for (auto i : l) cout << i << " "; cout << endl;
    for (auto &i : l) if (i < 3) i = 0;
    for (auto i : l) cout << i << " "; cout << endl;
    cout << "Valoare 0 se gaseste in lista de " << l[0] << " ori " << endl;
    cout << "Valoare 7 se gaseste in lista de " << l[7] << " ori " << endl;
    cout << "Lista are " << (int)l << " elemente" << endl;
    l -= 0;
    for (auto i : l) cout << i << " "; cout << endl;
    (l -= 3) -= 4;
    for (auto i : l) cout << i << " "; cout << endl;
    return 0;
}
```

să afișeze pe ecran următoarele:

```
9 7 5 3 1
8 6 4 2 9 7 5 3 1
3 2 1 8 6 4 2 9 7 5 3 1
3 0 0 8 6 4 0 9 7 5 3 0
Valoare 0 se gaseste in lista de 4 ori
Valoare 7 se gaseste in lista de 1 ori
Lista are 12 elemente
3 8 6 4 9 7 5 3
8 6 9 7 5
```

Observatii:

- Clasa SimpleList descrie o lista simplu inlantuita templetizata.
- **NU aveți** voie să utilizați containere/adaptori specifice *STL* (vector, list, etc)
- Constructorul lui SimpleList primește ca si parametru o lista de initializare
- Operatorul += adauga elemente in lista
- Operatorul -= șterge toate elementele cu o anumita valoare din lista.

Barem:

- **[3p]** Codul din template scris într-un fișier .h. Codul din *main* este într-un fișier separat.
- **[2p]** Constructorul pentru clasa SimpleList
- **[3p]** operatorul de cast la int
- **[4p]** operatorul de indexare
- **[4p]** operator += (cate 2 pct pentru fiecare forma)
- **[8p]** operator -=
- **[4p]** funcțiile necesare din clasa SimpleList pentru funcționarea for-each-ului
- **[7p]** iteratorul pentru for-each