1. **AgendaP**

Acest cod creeaza o fereastra grafica prin intermediul bibliotecii Swing din Java. "JFrame" este o clasa din biblioteca Swing care permite crearea de ferestre. In acest cod, se creeaza o noua instanta a acestei clase, numita "w", care va fi folosita pentru a afisa fereastra.

Cu ajutorul functiilor "setSize" si "setLocation", se stabilesc dimensiunile si pozitia ferestrei pe ecran. Functia "setVisible" permite afisarea ferestrei pe ecran, iar functia "setDefaultCloseOperation" stabileste ce se intampla cand se inchide fereastra (in acest caz, se termina programul).

1. **AgendaPreot**

Acest cod reprezintă o clasă Java numită "AgendaPreot", care gestionează evenimente bisericești. Clasa folosește colecția TreeSet pentru a stoca evenimentele și fișierul "evenimenteBisericesti.txt" pentru a salva și încărca evenimentele. Clasa oferă funcții precum adăugarea, căutarea, ștergerea și modificarea evenimentelor, precum și afișarea tuturor evenimentelor din agenda. Aceste operații se efectuează folosind obiecte precum BufferedReader și PrintWriter pentru citirea și scrierea în fișier. JOptionPane este utilizat pentru a afișa mesaje de eroare sau informații către utilizator.

1. **CautaEvenimentFrame**

Acest cod defineste o clasa Java numita "CautaEvenimentFrame", care extinde clasa "JFrame" din pachetul "javax.swing". Clasa creeaza o fereastra grafica cu un camp de text si o zona de text pentru afisare. Cand utilizatorul introduce text in campul de text si apasa Enter, se apeleaza metoda "actionPerformed" din clasa interna "GestorEvenimenteCFF". In aceasta metoda, se cauta un eveniment in agenda folosind metoda "cautaEveniment" a obiectului "AgendaPreot". Rezultatul cautarii este afisat in zona de text.

1. **Eveniment**

Acest cod defineste o clasa Java numita "Eveniment". Clasa are multiple proprietati si metode pentru a stoca si a obtine informatii despre un eveniment, cum ar fi numele, prenumele, adresa, numarul de telefon, tipul evenimentului, data si ora.

Metodele din clasa permit setarea si obtinerea acestor proprietati, precum si obtinerea unui String formatat care descrie evenimentul.

Clasa implementeaza interfata "Comparable" si suprascrie metoda "compareTo", care permite compararea a doua obiecte Eveniment in functie de numele si prenumele lor.

1. **EvenimentNouFrame**

Acest cod creaza o fereastra Java Swing pentru adaugarea unui nou eveniment in agenda unui preot. Ea contine un panou cu campuri pentru a introduce informatii despre eveniment, cum ar fi numele, prenumele, adresa, data, ora, numarul de telefon si tipul evenimentului.

Exista, de asemenea, doua butoane - "OK" si "Cancel" - care au un ascultator de evenimente asociat. Daca utilizatorul apasa pe butonul "OK", se adauga un nou eveniment in agenda prin apelarea metodei "adaugaEveniment" din clasa "AgendaPreot". Daca utilizatorul apasa pe butonul "Cancel", fereastra se inchide.

1. **EvenimenteFrame**

Acest cod creeaza o clasa numita "EvenimenteFrame" care este o fereastra de aplicatie Java Swing. Clasa extinde clasa JFrame. In interiorul clasei, se declara un obiect JTextArea numit "ta" si un obiect AgendaPreot numit "ap".

Constructorul clasei creeaza o fereastra cu titlul "Agenda preot" si adauga un JLabel cu textul "Toate evenimentele" in partea de sus a ferestrei.

Textul din JTextArea "ta" este setat la rezultatul apelarii metodei "getEvenimente" din obiectul "ap" de tip AgendaPreot. JTextArea este setata sa fie needitabila prin setarea proprietatii "setEnabled" la "false".

Clasa se inchide cu adaugarea unui ascultator de fereastra care se ocupa de inchiderea ferestrei atunci cand se apasa pe butonul de inchidere.

1. **FereastraPrincipala**

Acest cod implementeaza o clasa "FereastraPrincipala" care extinde clasa JFrame din pachetul "javax.swing". Aceasta clasa are un array de butoane "b" si un obiect al clasei interne "GestorEvenimenteFP", care implementeaza interfata "ActionListener".

Fiecare buton are un listener asociat prin intermediul metodei "addActionListener()", care apelaza metoda "actionPerformed()" din clasa "GestorEvenimenteFP" atunci cand un buton este apasat.

In metoda "actionPerformed()", se verifica sursa evenimentului si se deschide fereastra corespunzatoare (fie "EvenimentNouFrame", "CautaEvenimentFrame", "StergeEvenimentFrame", "EvenimenteFrame", "ModificaEvenimentFrame", "ListaSfinti", "PredicaAnunt", etc) in functie de butonul apasat.

De asemenea, metoda "actionPerformed()" poate salva toate evenimentele din agenda prin apelarea metodei "salveaza()" din clasa AgendaPreot.

1. **ModificaEvenimentFrame**

Acest cod definește o clasă numită "ModificaEvenimentFrame" care extinde JFrame și care implementează o interfață grafică cu ajutorul bibliotecii Swing din Java. Clasa conține câmpuri de tip JTextField și JTextArea, precum și un buton JButton pentru modificarea unui eveniment. Când utilizatorul apasă butonul "Modifica", se apelază metoda "modificaEveniment" din clasa "AgendaPreot" cu valorile introduse în câmpurile JTextField. Clasa "ModificaEvenimentFrame" conține, de asemenea, o clasă internă numită "GestorEvenimenteCFF" care implementează interfața ActionListener și care gestionează acțiunea asociată butonului "Modifica".

1. **StergeEvenimentFrame**

Acest cod defineste clasa "StergeEvenimentFrame" care este o clasa de tip JFrame (fereastra). Aceasta clasa afiseaza un formular cu doua butoane "OK" si "Cancel" si un textfield. Utilizatorul poate introduce nume si prenume in textfield. Butonul "OK" foloseste informatia din textfield pentru a sterge un eveniment din agenda prin intermediul metodei "stergeEveniment". Butonul "Cancel" inchide fereastra. Clasa "GestorEvenimenteFF" este o clasa interna si implementeaza interfata "ActionListener" si are rolul de a gestiona actiunile utilizatorului prin intermediul butoanelor.

1. **ListaSfinti**

Acest cod reprezinta o aplicatie Java care afiseaza numele unui sfant pentru o anumita data. Acesta utilizeaza clasa Calendar din pachetul java.util pentru a gestiona data si clasa JFrame din pachetul javax.swing pentru a crea fereastra aplicatiei. Aplicatia definește un map de numere la șiruri care conține zilele lunii februarie asociate cu numele sfintilor si o metoda isSfant() care verifica daca o anumita data este asociata cu un sfant. De asemenea, exista o metoda getNumeSfant() care returneaza numele sfantului asociat cu o anumita data. In plus, aceasta clasa defineste o fereastra GUI care permite utilizatorului sa introduca o luna si o zi si afiseaza numele sfantului corespunzator sau mesajul "Nu este un sfant" daca nu exista asociere pentru acea data.

1. **PredicaAnunt**

Acest cod implementeaza o aplicatie Java Swing care permite utilizatorilor sa adauge, afiseze, modifice si sa stearga predici sau anunturi. Când se apasă pe butonul "Adauga", utilizatorul poate introduce o noua predica/anunt prin intermediul unui fereastra de dialog, care va fi adaugat apoi la sfarsitul listei de predici/anunturi existente. Când se apasă pe butonul "Afiseaza predici/anunturi", lista de predici/anunturi va fi incarcata din fisierul "notes.txt" si afisata în zona de text. Butoanele "Sterge" si "Modifica" permit utilizatorilor sa stearga sau sa modifice o predica/anunt existent, tot prin intermediul unui fereastra de dialog. Toate operatiunile de adaugare, afisare, modificare si stergere sunt salvate in fisierul "notes.txt".