

Algoritmi de Sortări



Autor:

Adrian-Iulian Mitrofan-Bîtcă

Structura prezentării

1. De ce?
2. Pași simpli de urmat
3. Aspecte grafice
4. Funcționare obiecte
5. Programare Java
6. Testarea aplicației
7. Concluzii

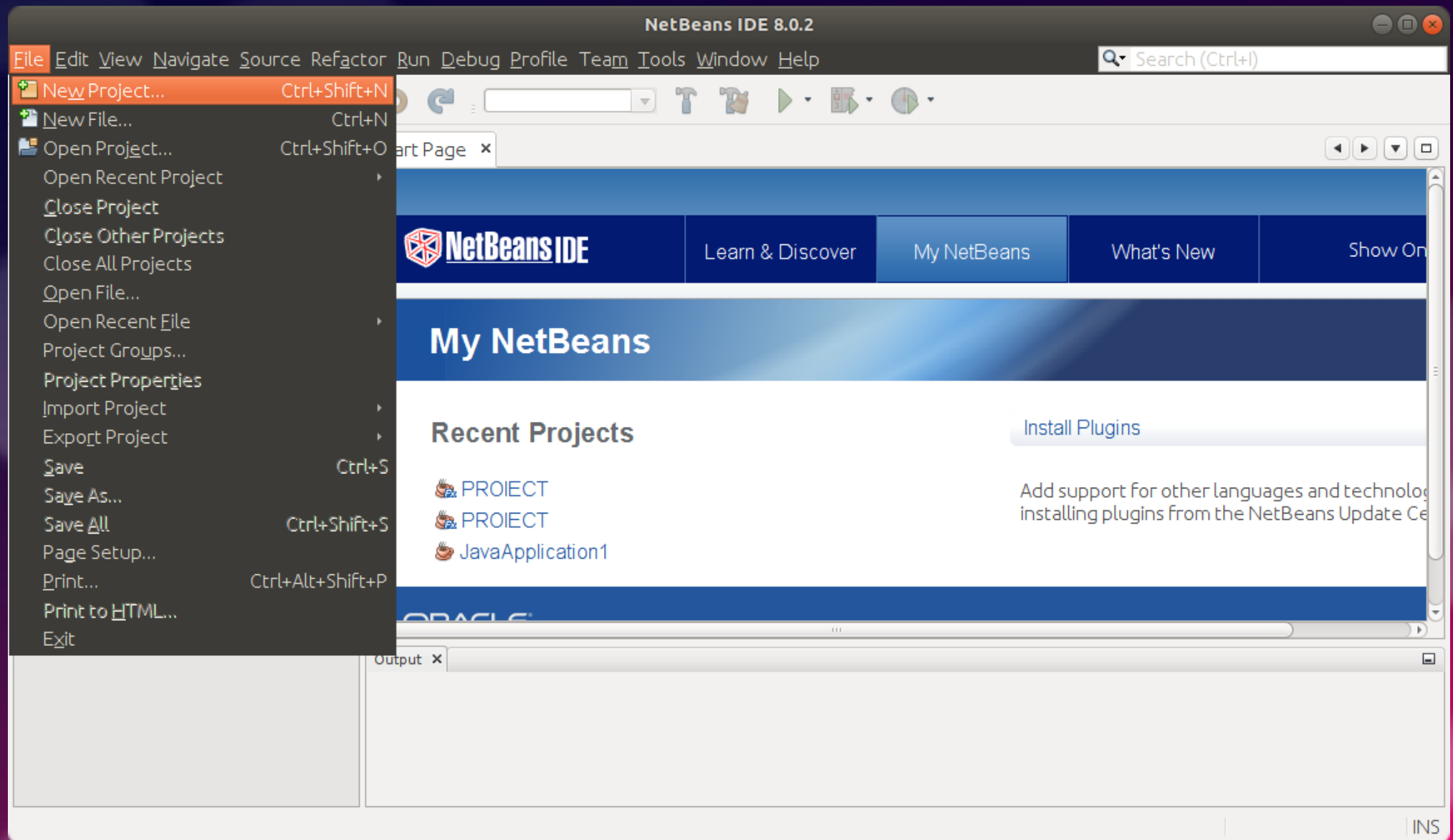


1. De ce?

- De ce sortări?
- Aplicarea în practică a unor concepte teoretice
 - De ce NetBeans, JavaFx și Swing?
- În câțiva pași simpli poți ajunge la rezultate vizibile, datorită librăriilor grafice, care ajută dezvoltatorul să se concentreze mai mult pe partea de programare a aplicației

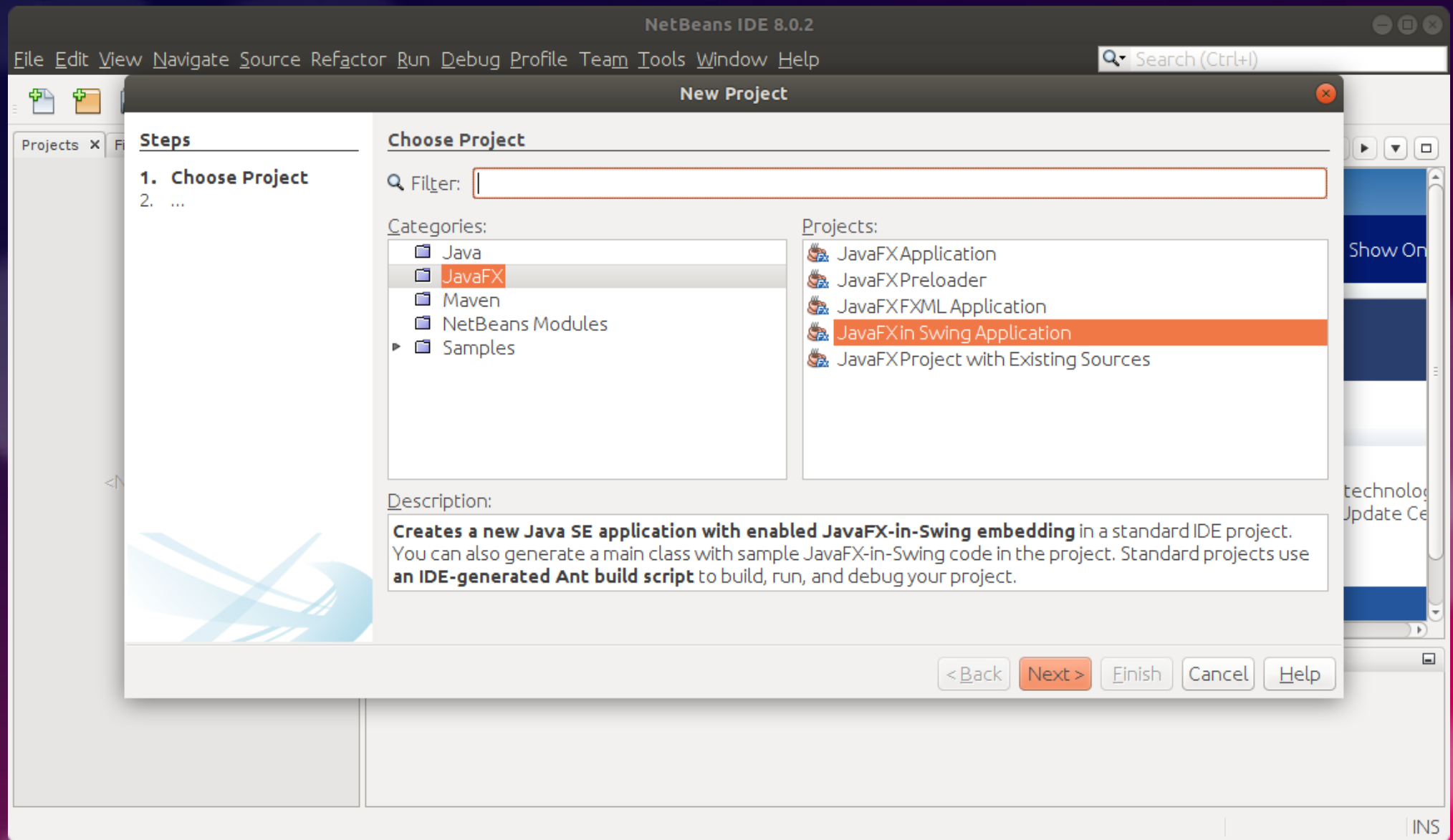
2. Pași simpli de urmat

a. Un nou proiect

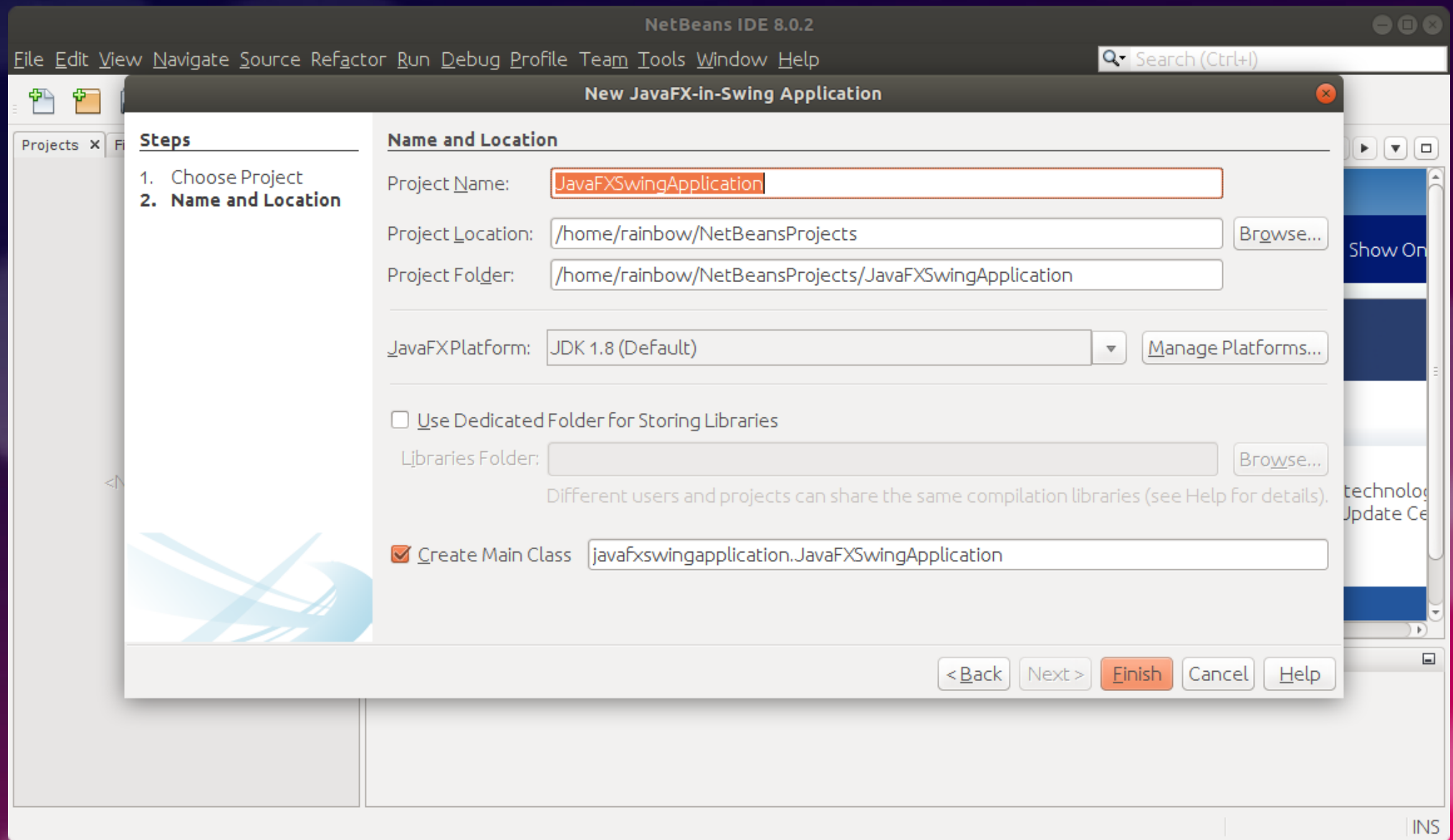


2. Pași simpli de urmat

b. Aplicație JavaFX cu obiecte Swing

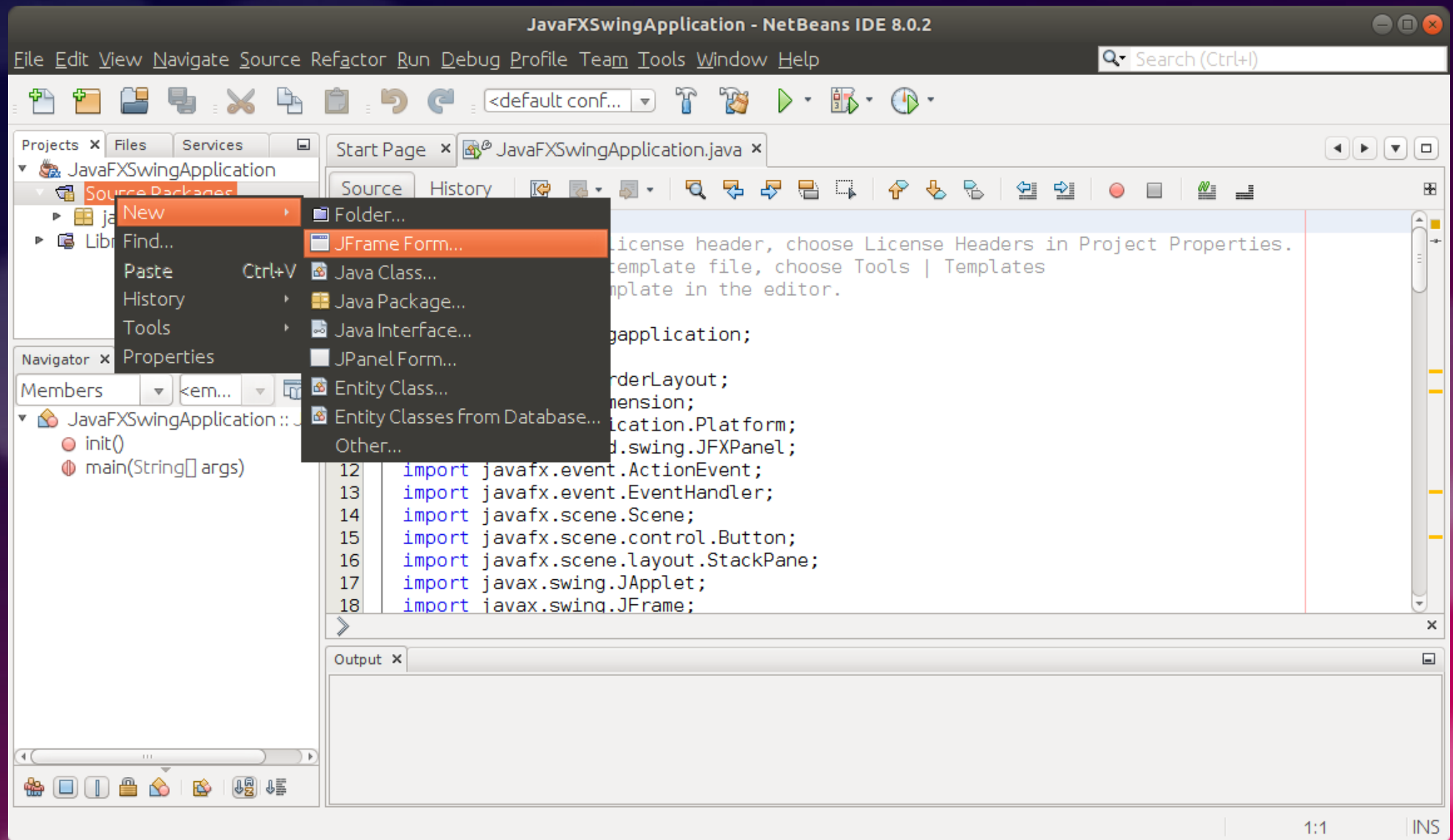


2. Pași simpli de urmat c. Platforma de lucru

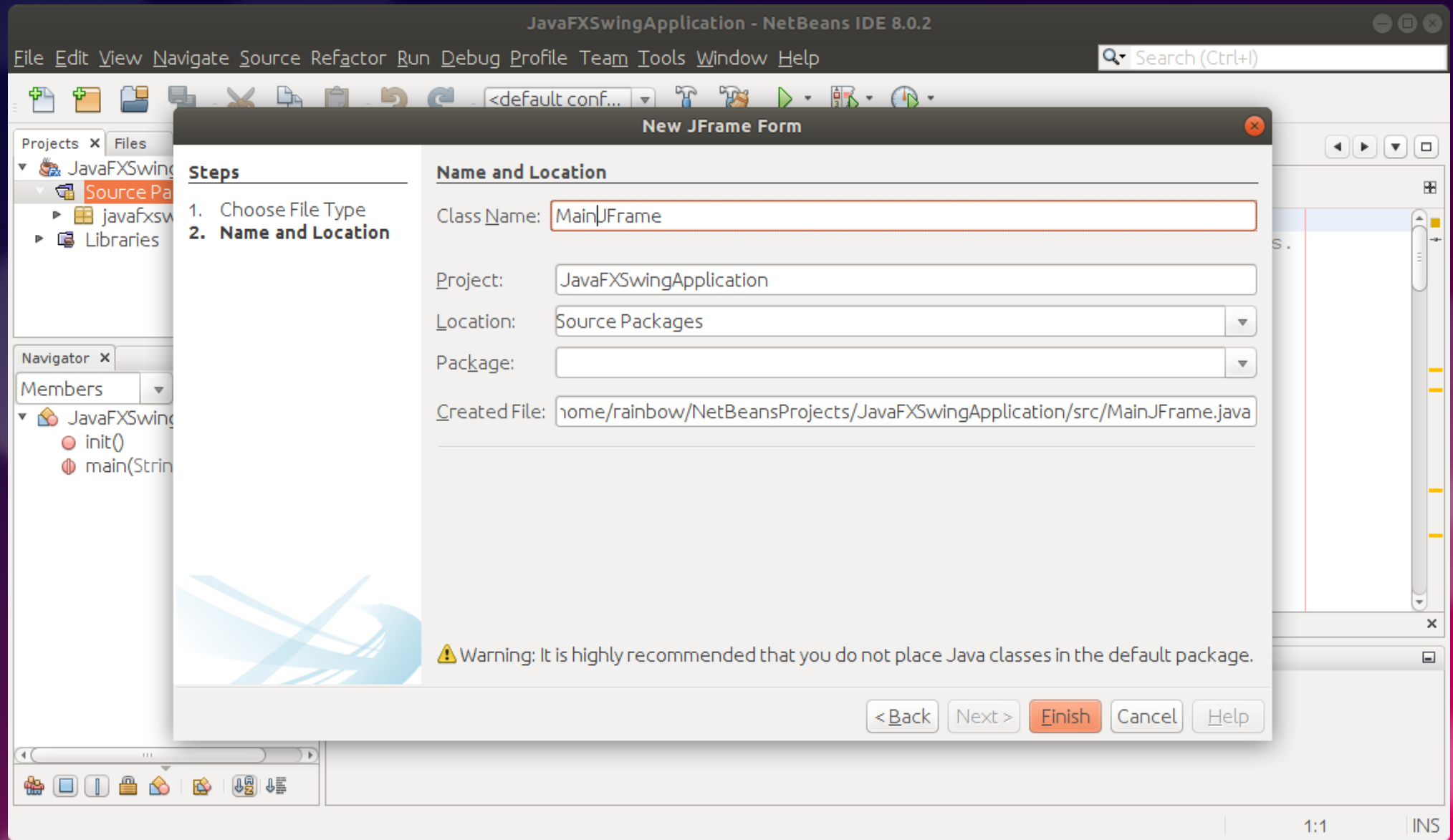


2. Pași simpli de urmat

d. Inserarea cadrului de lucru

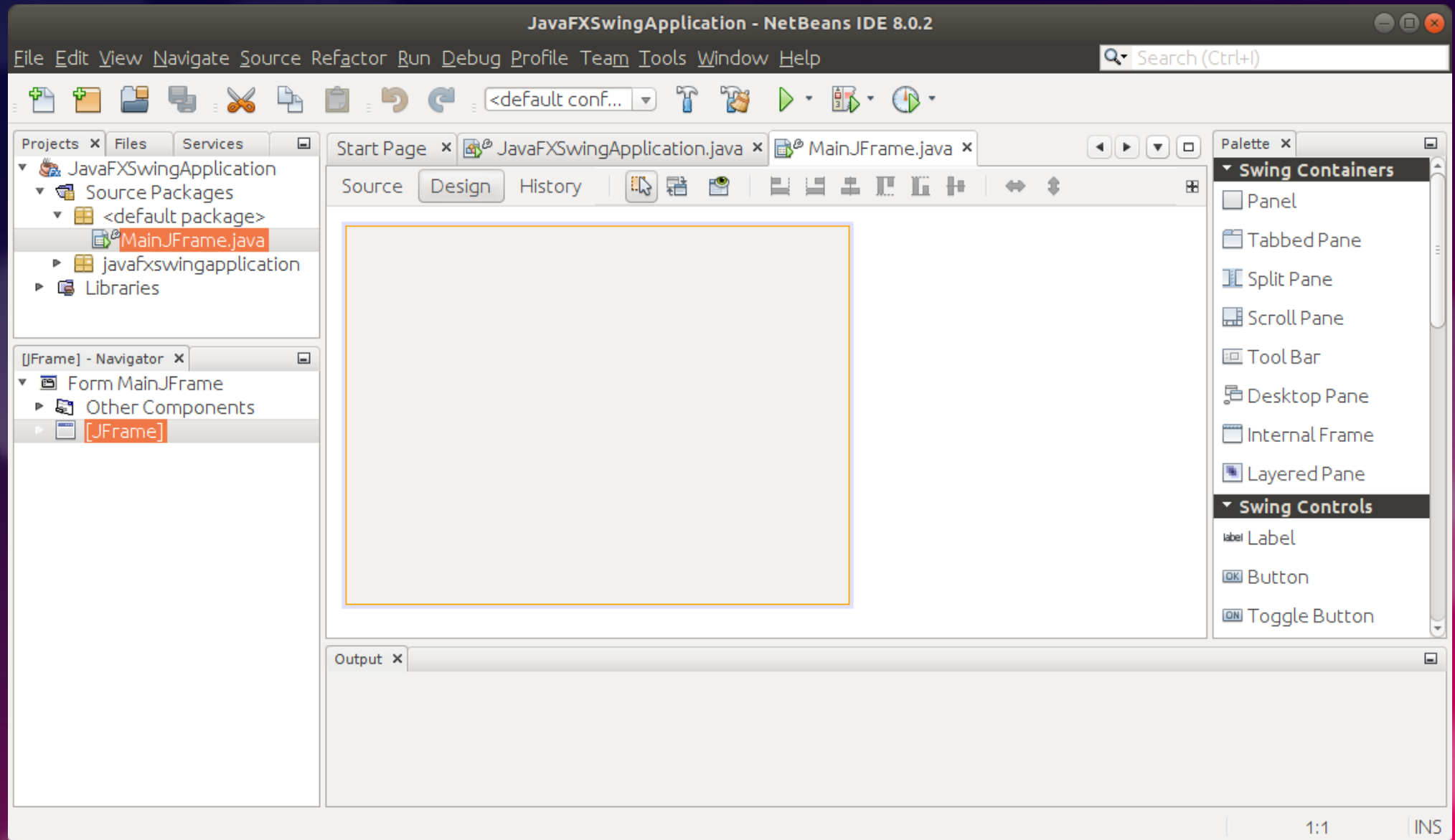


2. Pași simpli de urmăriți e. Denumirea cadrului de lucru



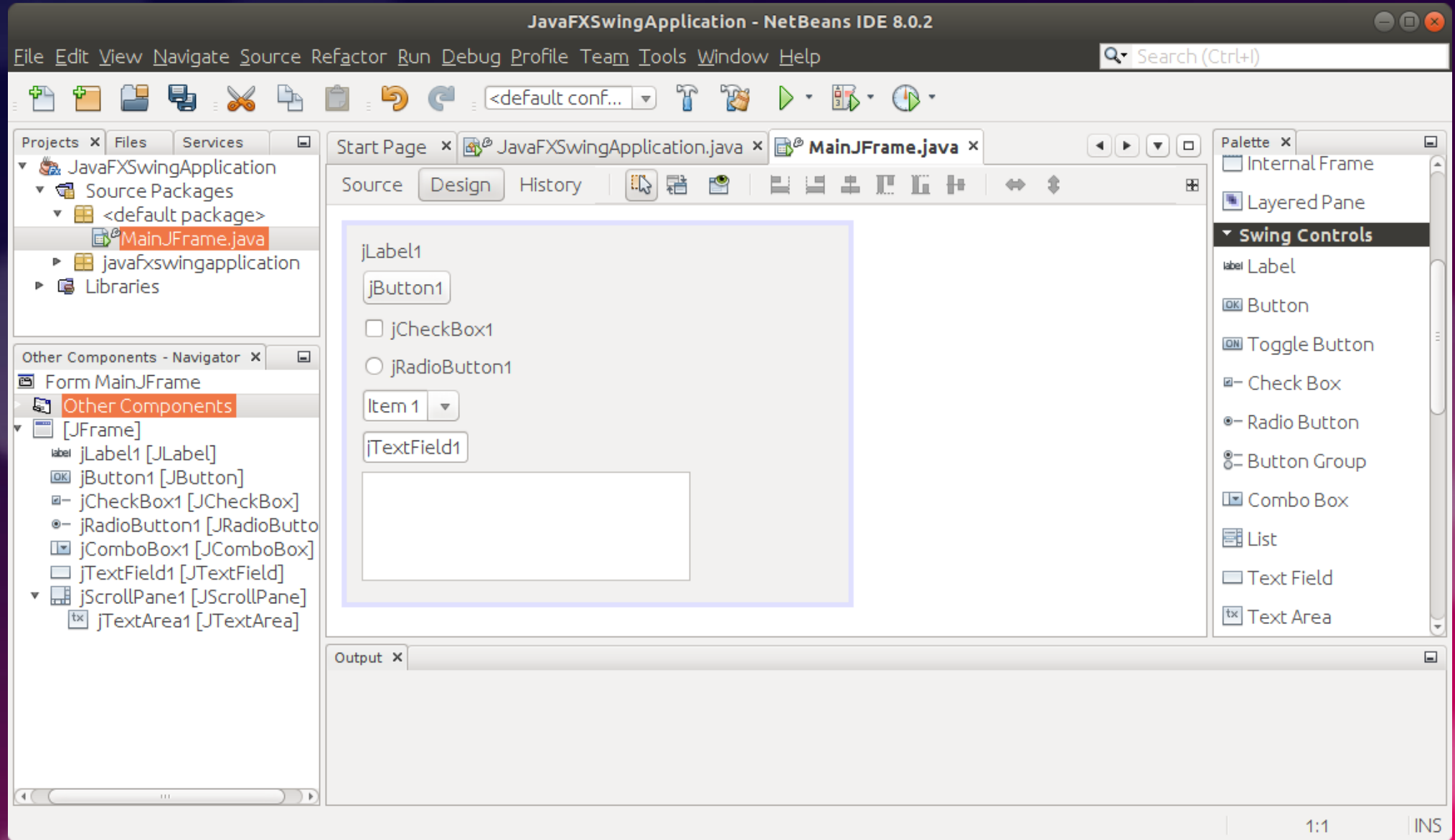
3. Aspecte grafice

a. Alegerea componentelor



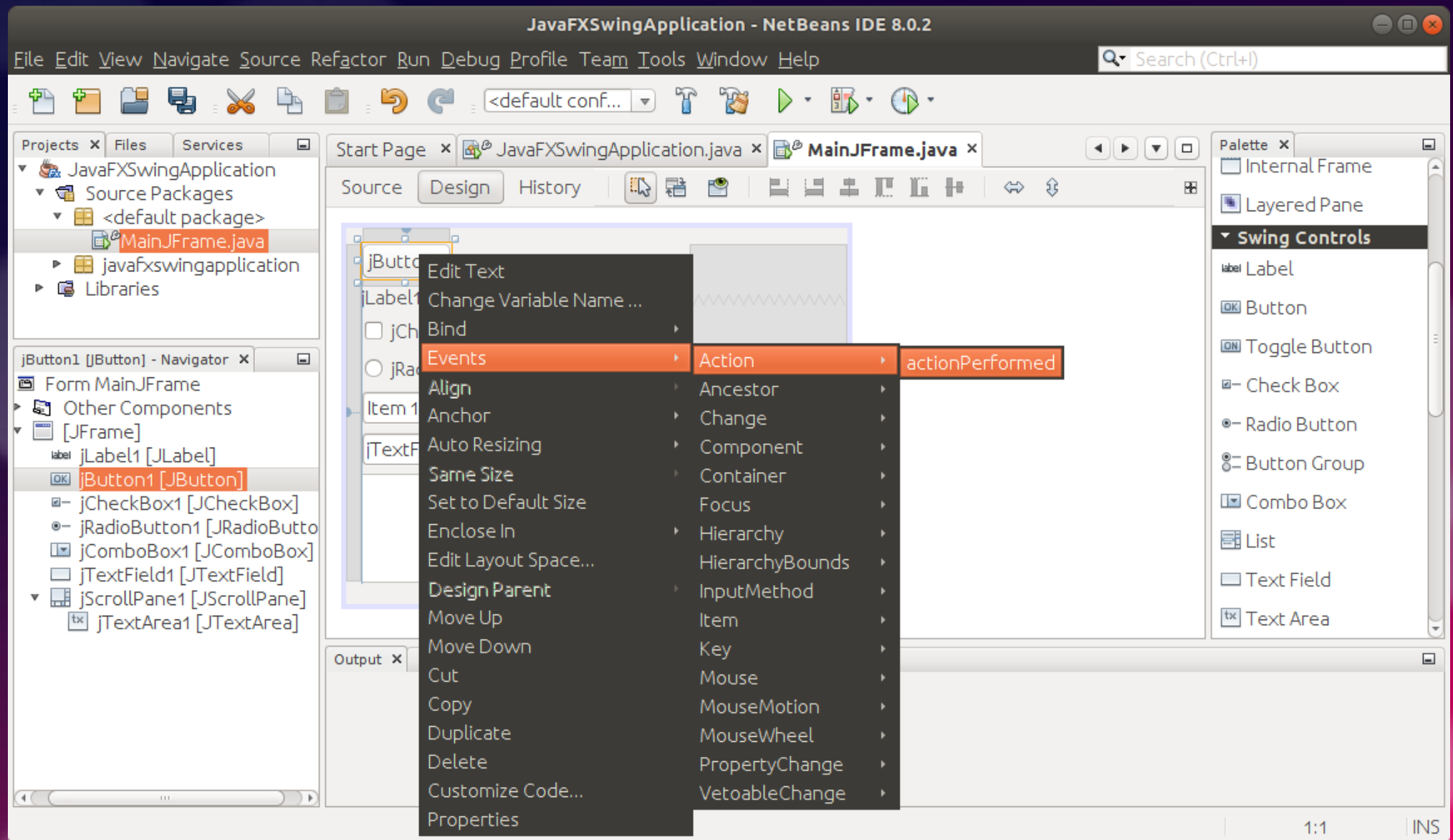
3. Aspecte grafice

b. Obiecte necesare



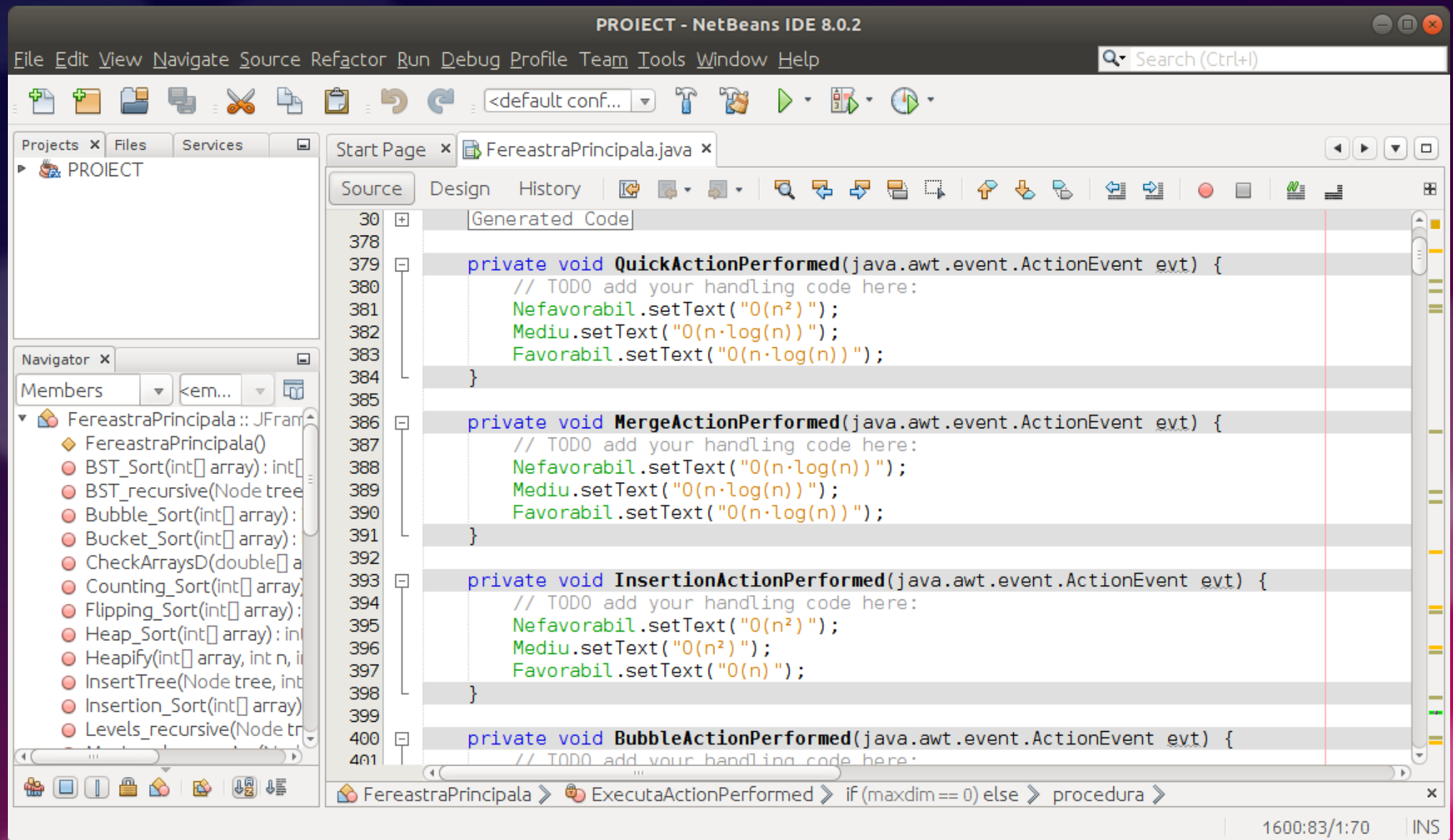
4. Funcționare obiecte

a. Obiecte și evenimente



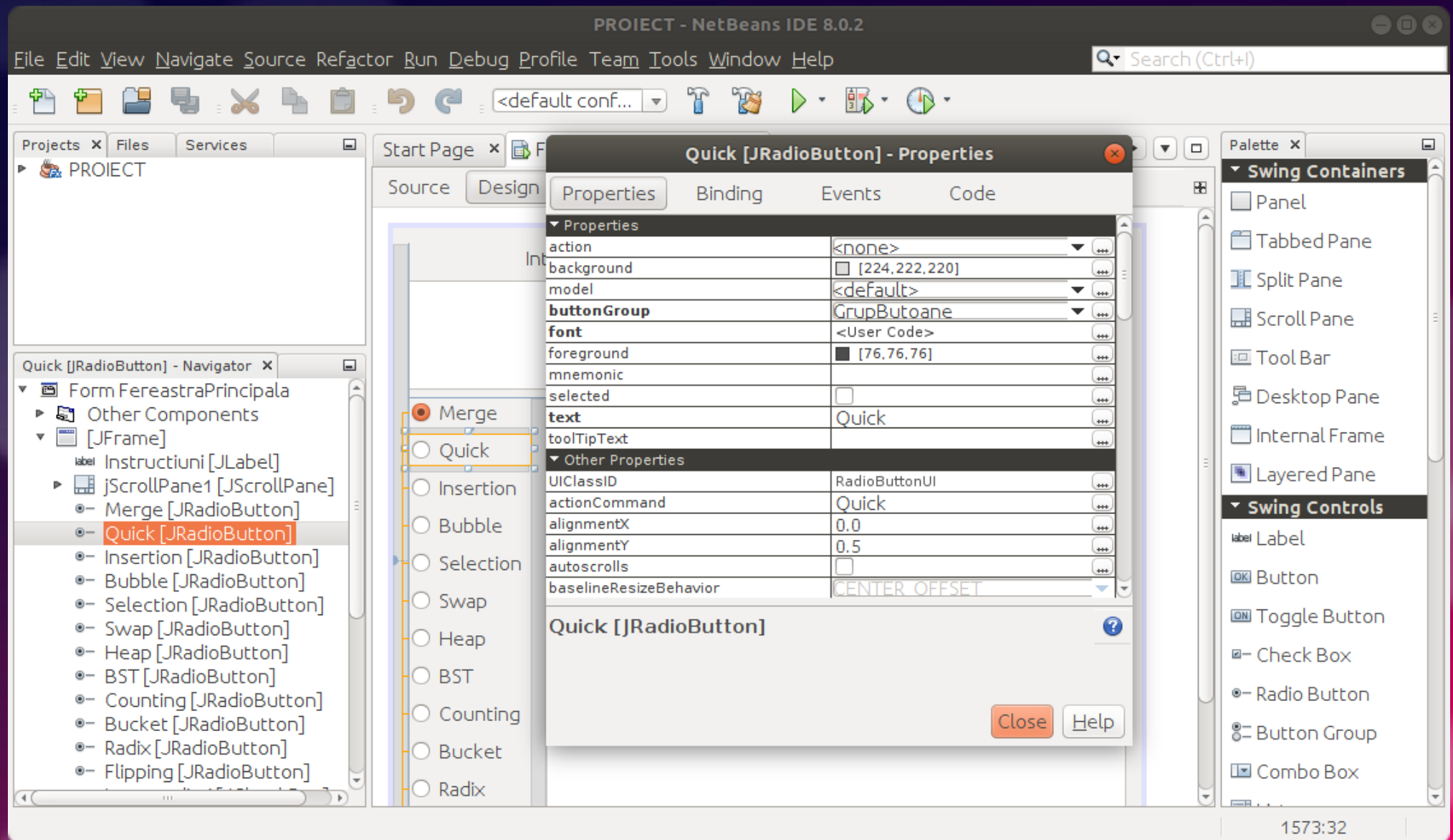
4. Funcționare obiecte

b. Butoane radio și selectarea lor



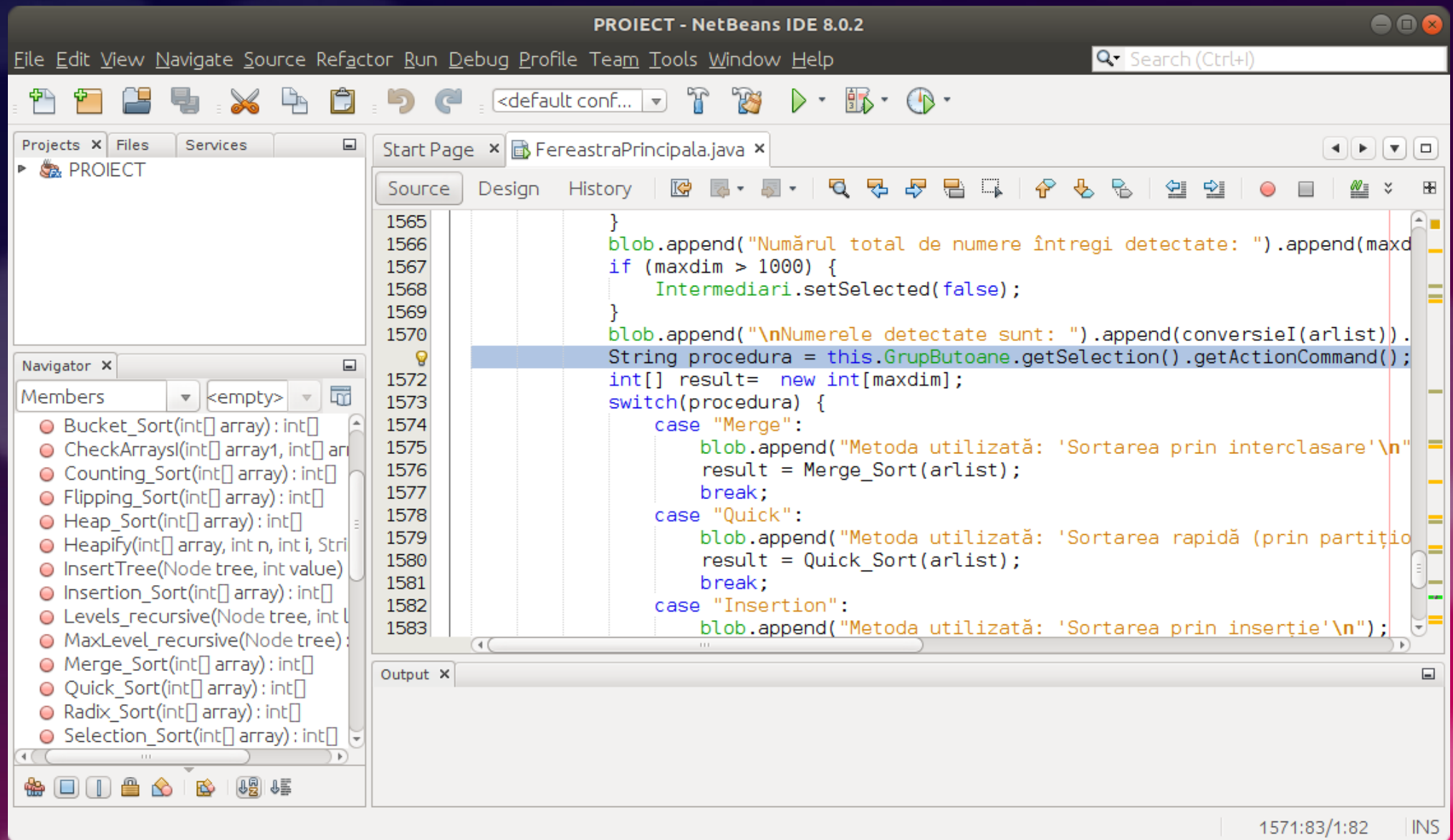
4. Funcționare obiecte

c. Grupuri de butoane



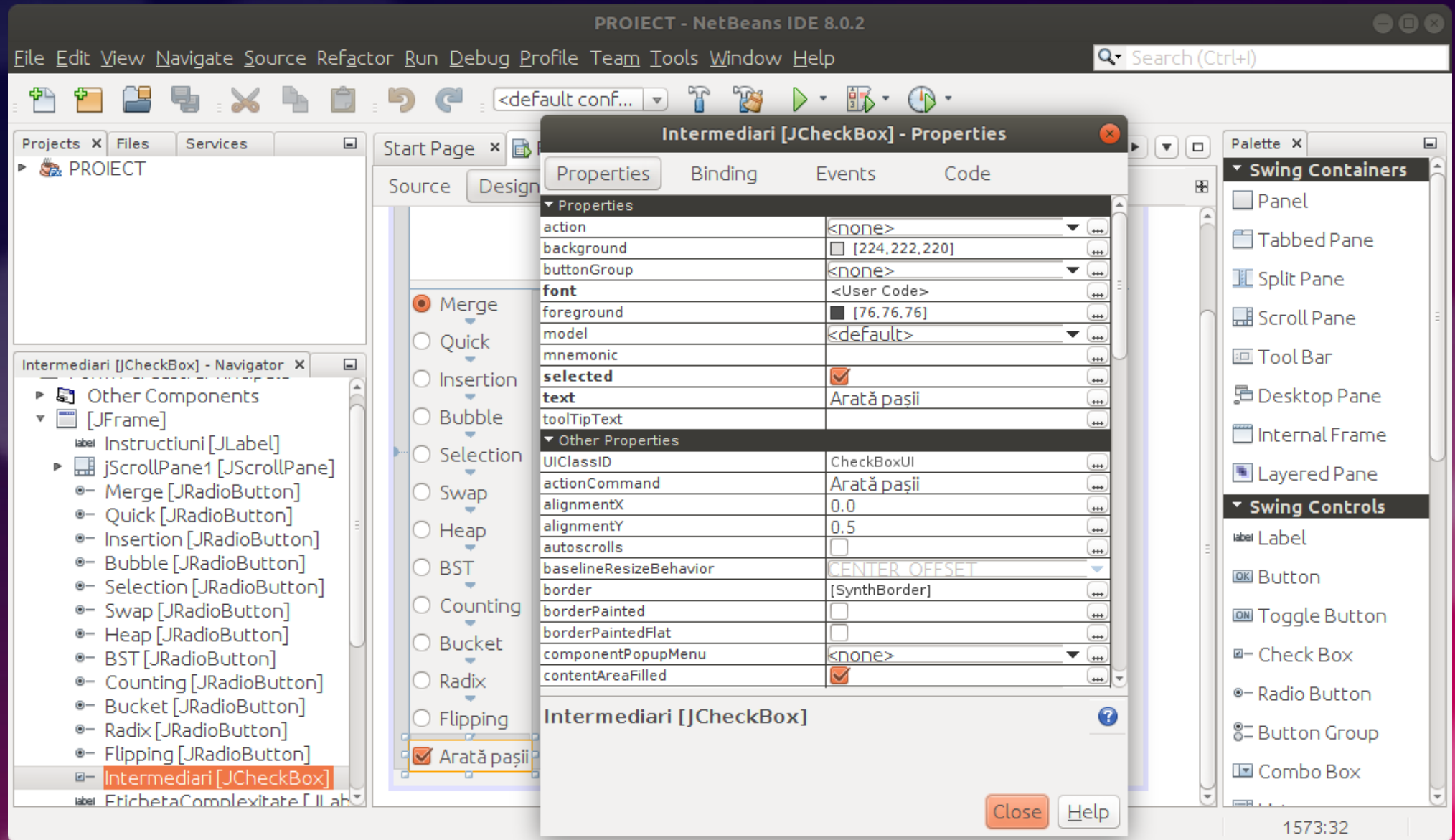
4. Funcționare obiecte

d. Atribute din grup



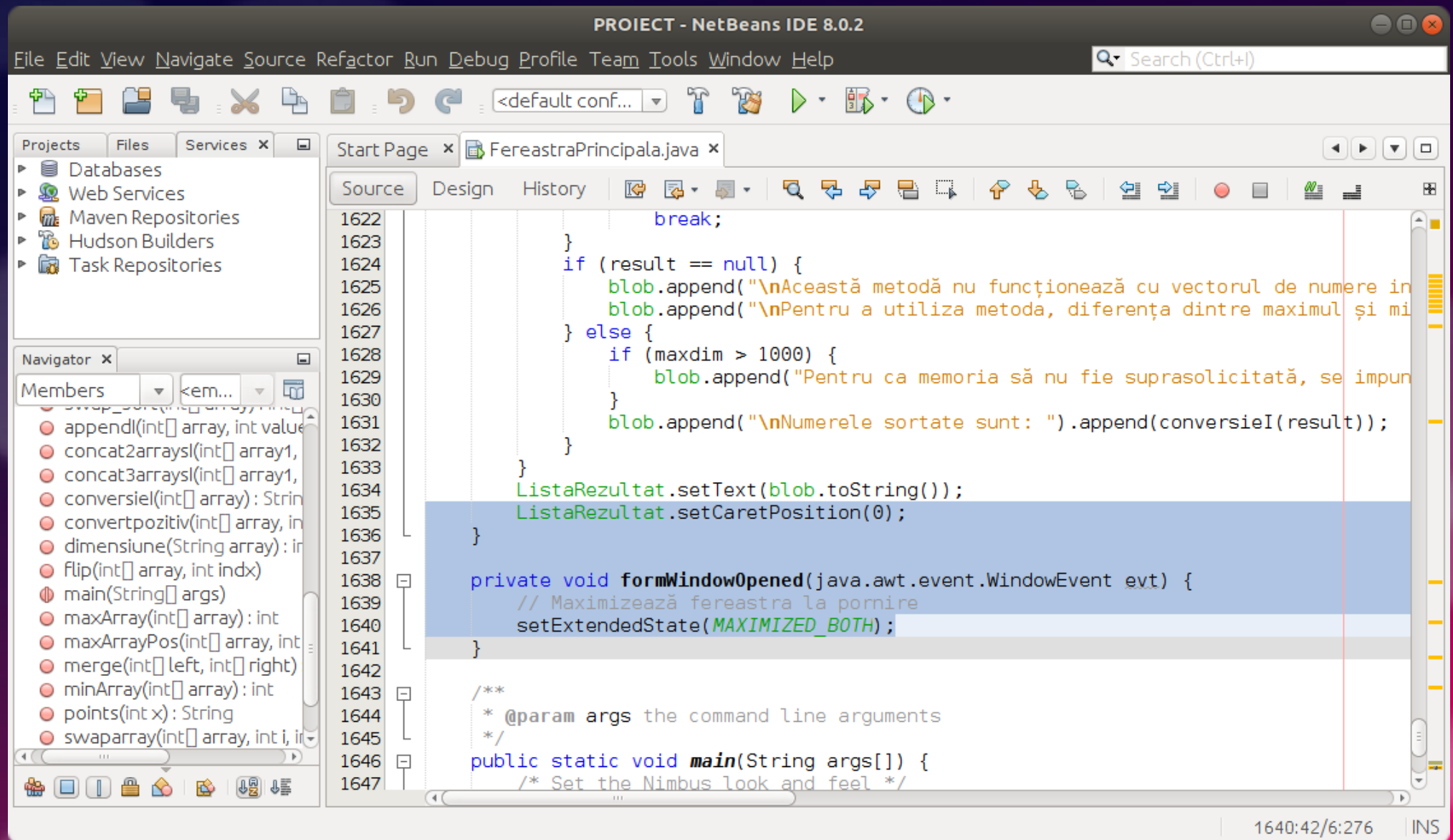
4. Funcționare obiecte

e. Checkbox



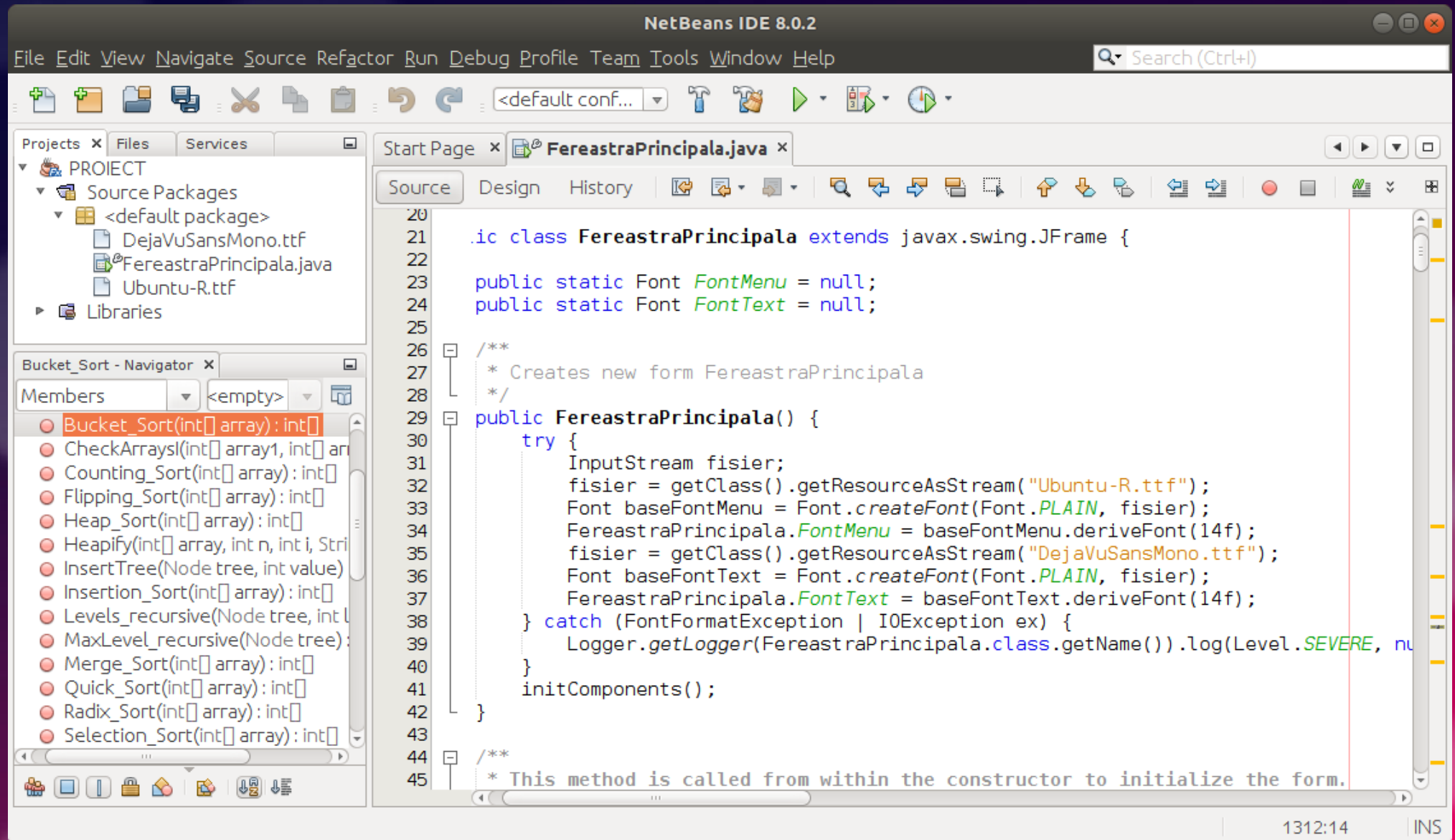
4. Funcționare obiecte

f. Poziționări



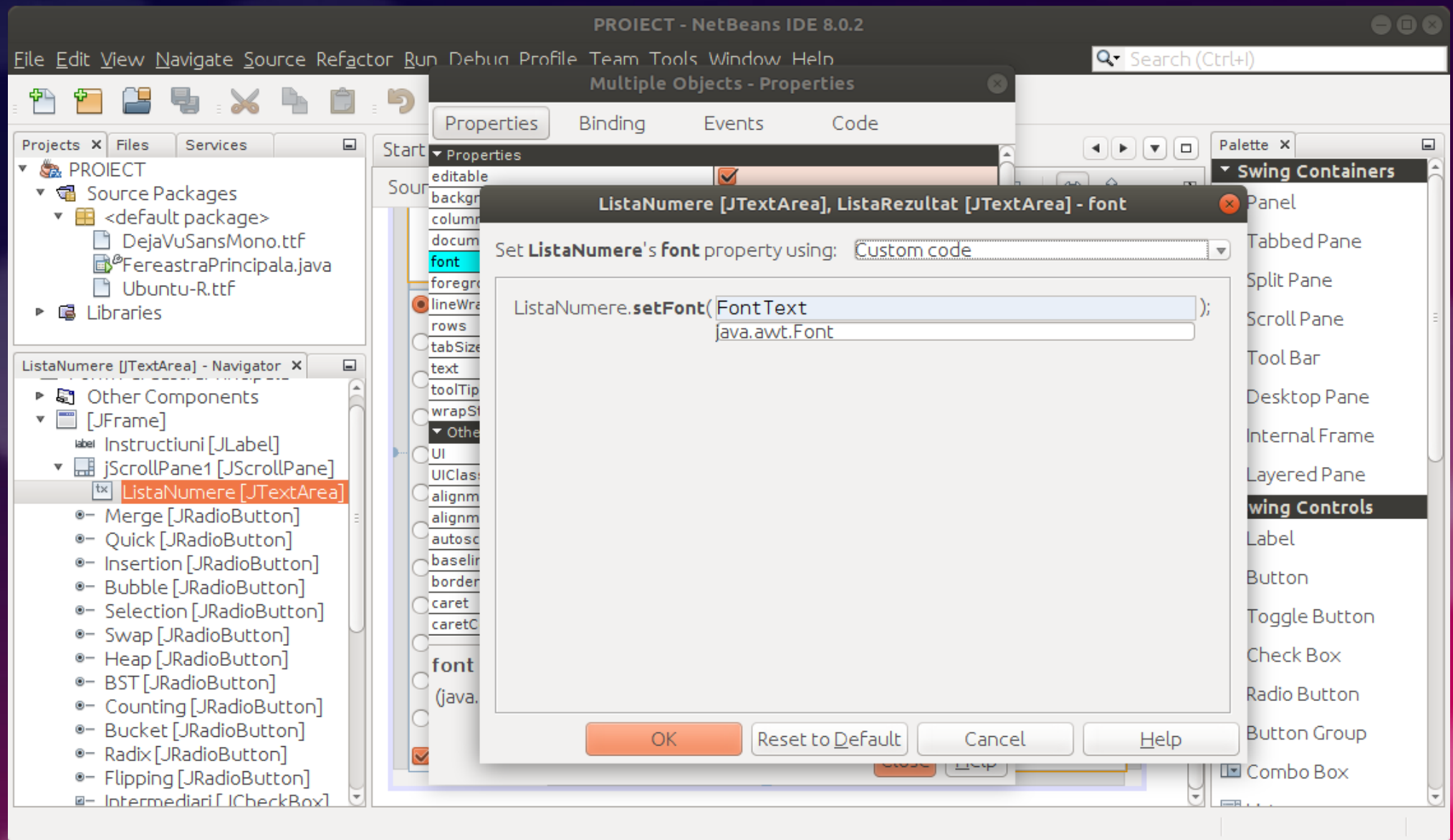
4. Funcționare obiecte

g. Inserare fonturi în aplicație



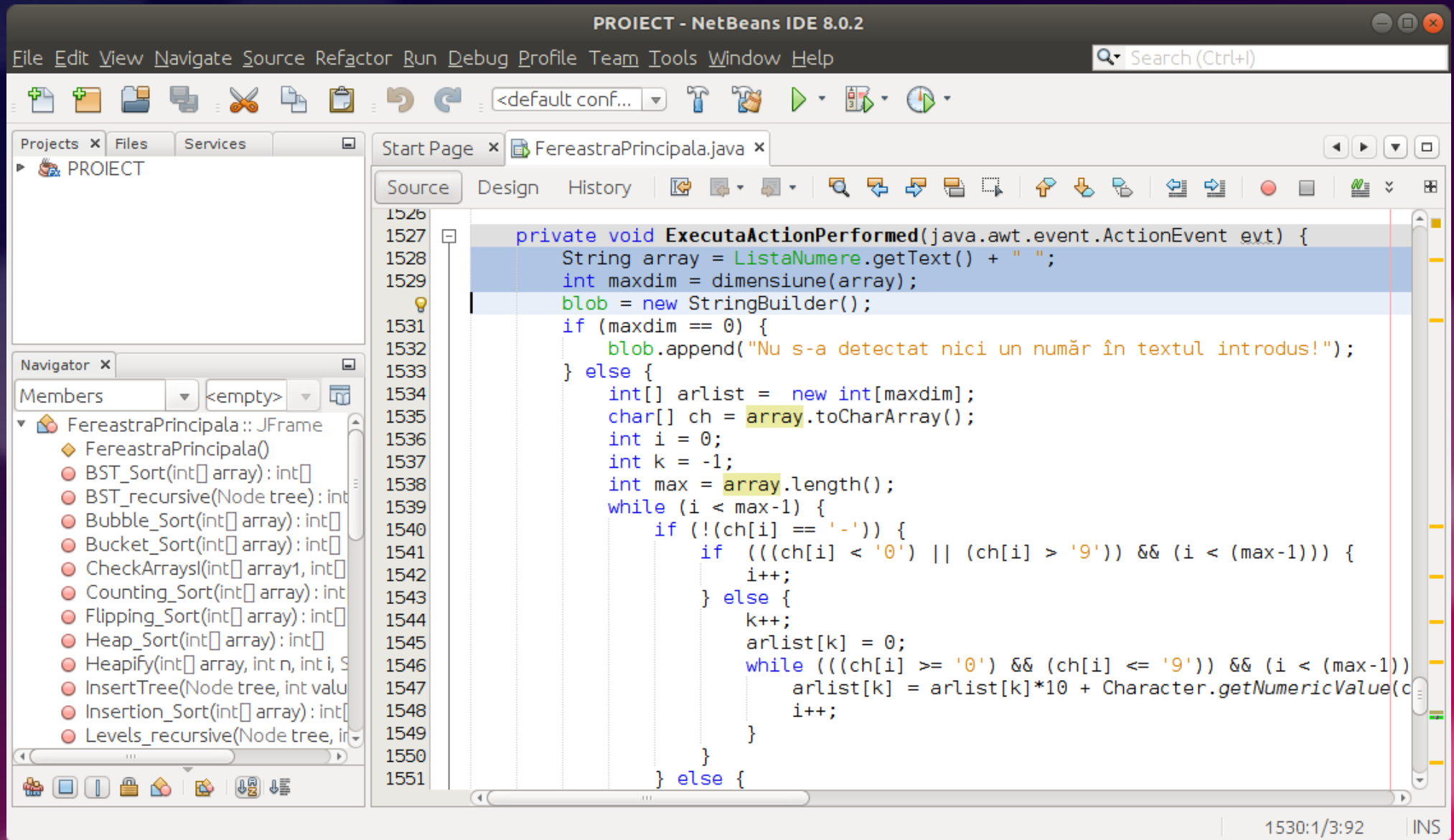
4. Funcționare obiecte

h. Utilizarea fonturilor



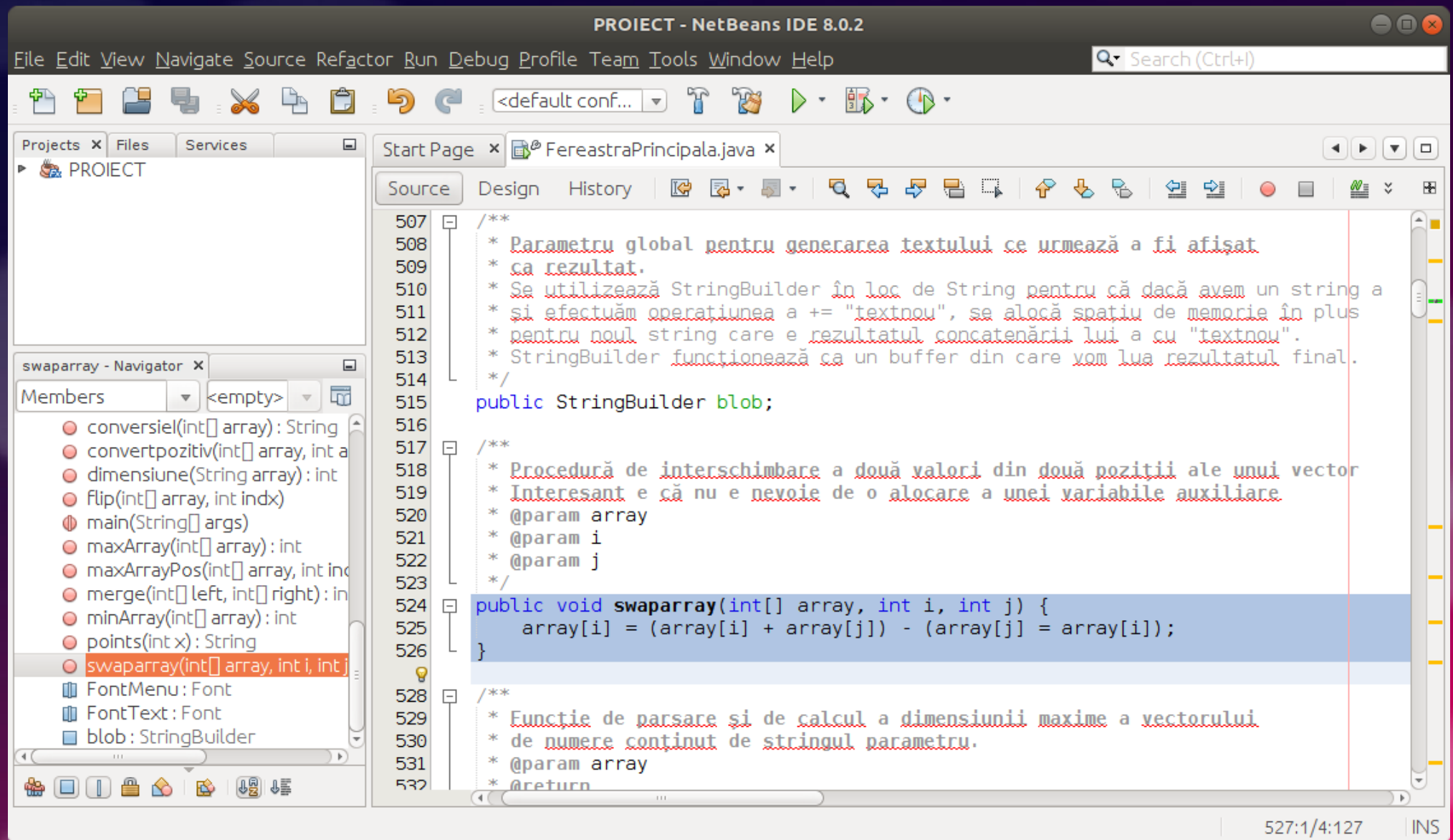
5. Programare Java

a. Date de intrare



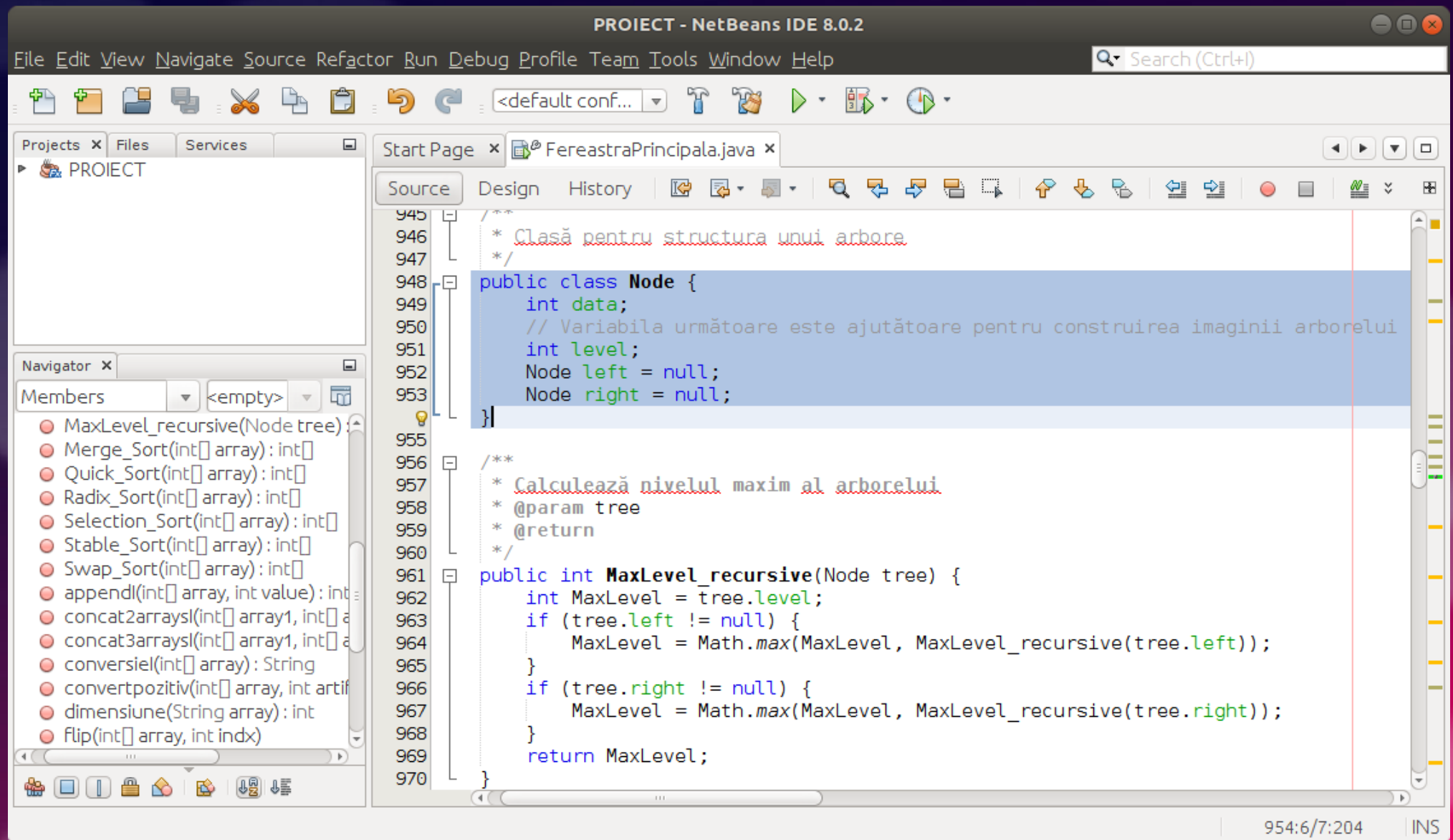
5. Programare Java

b. Variabile și proceduri



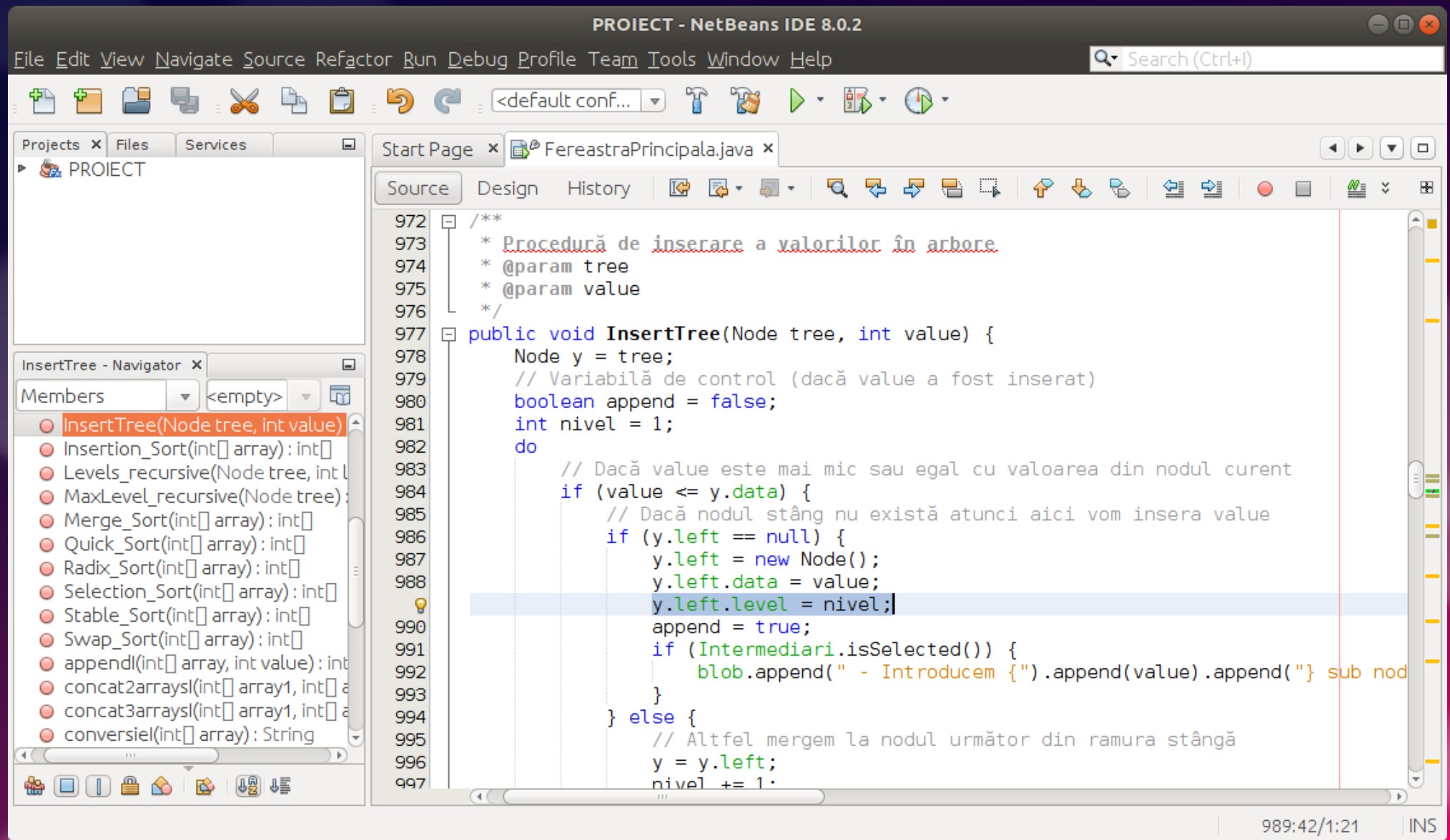
5. Programare Java

c. Structuri necesare



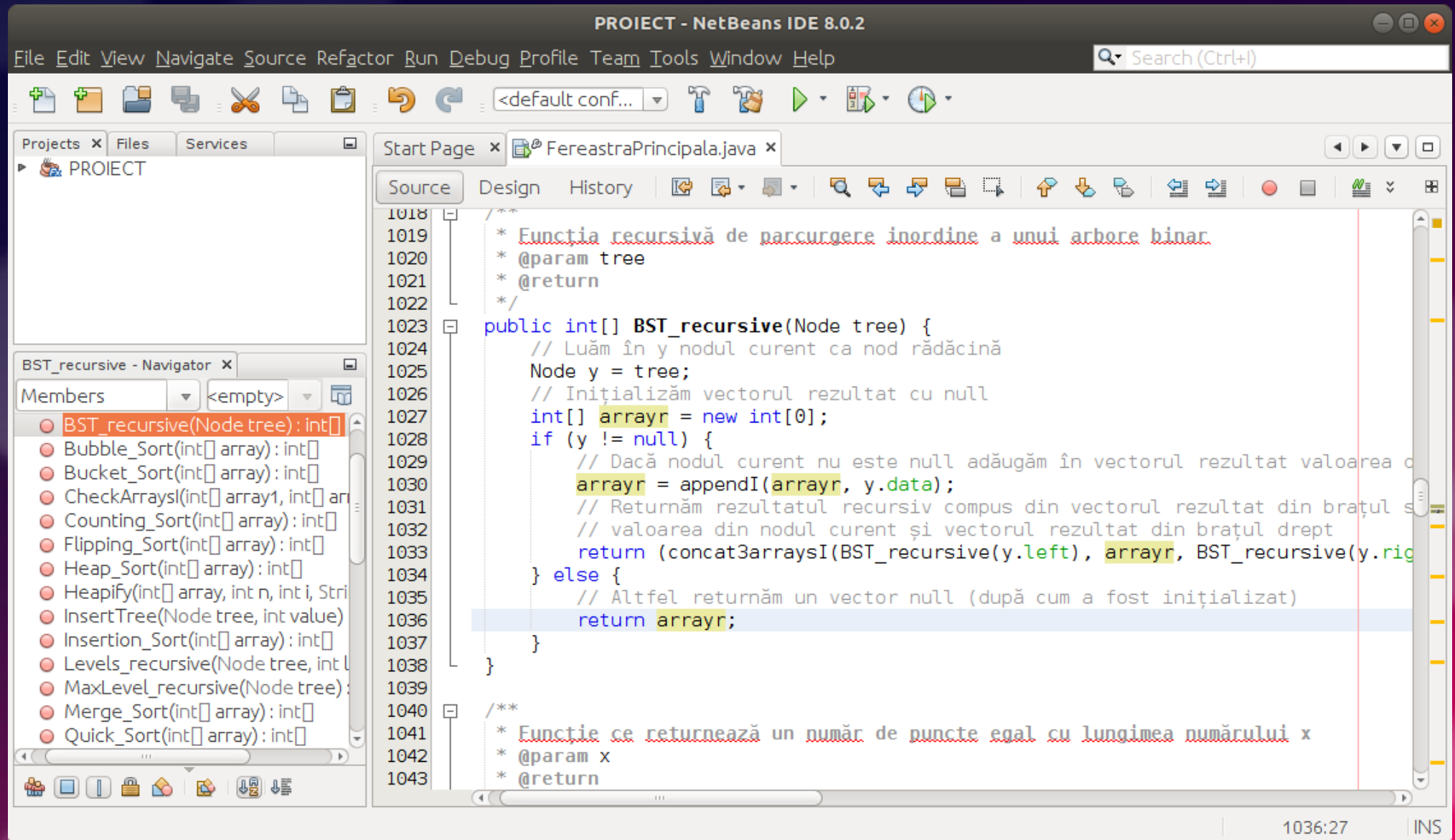
5. Programare Java

d. Popularea unui arbore



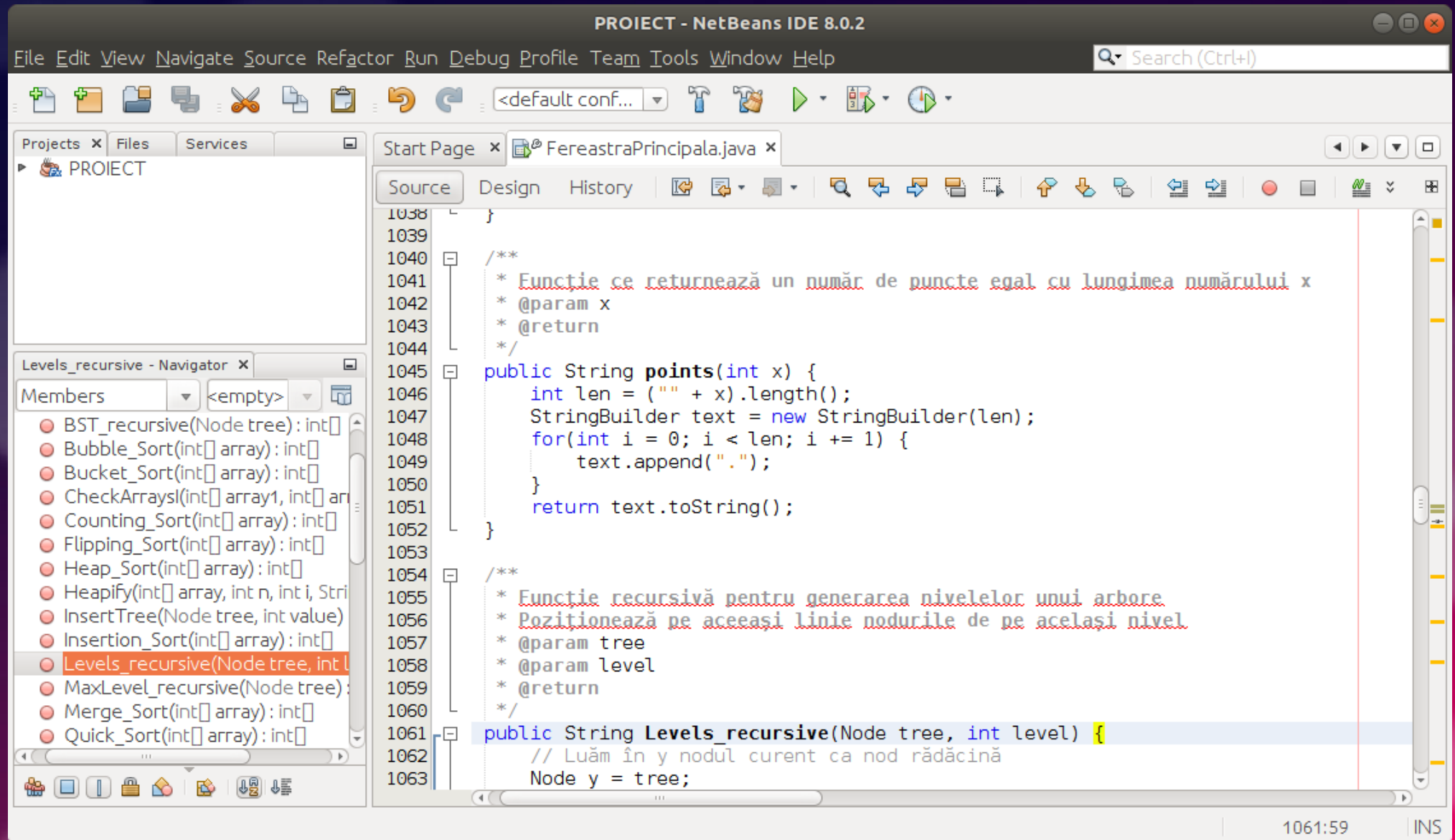
5. Programare Java

e. Funcții recursive



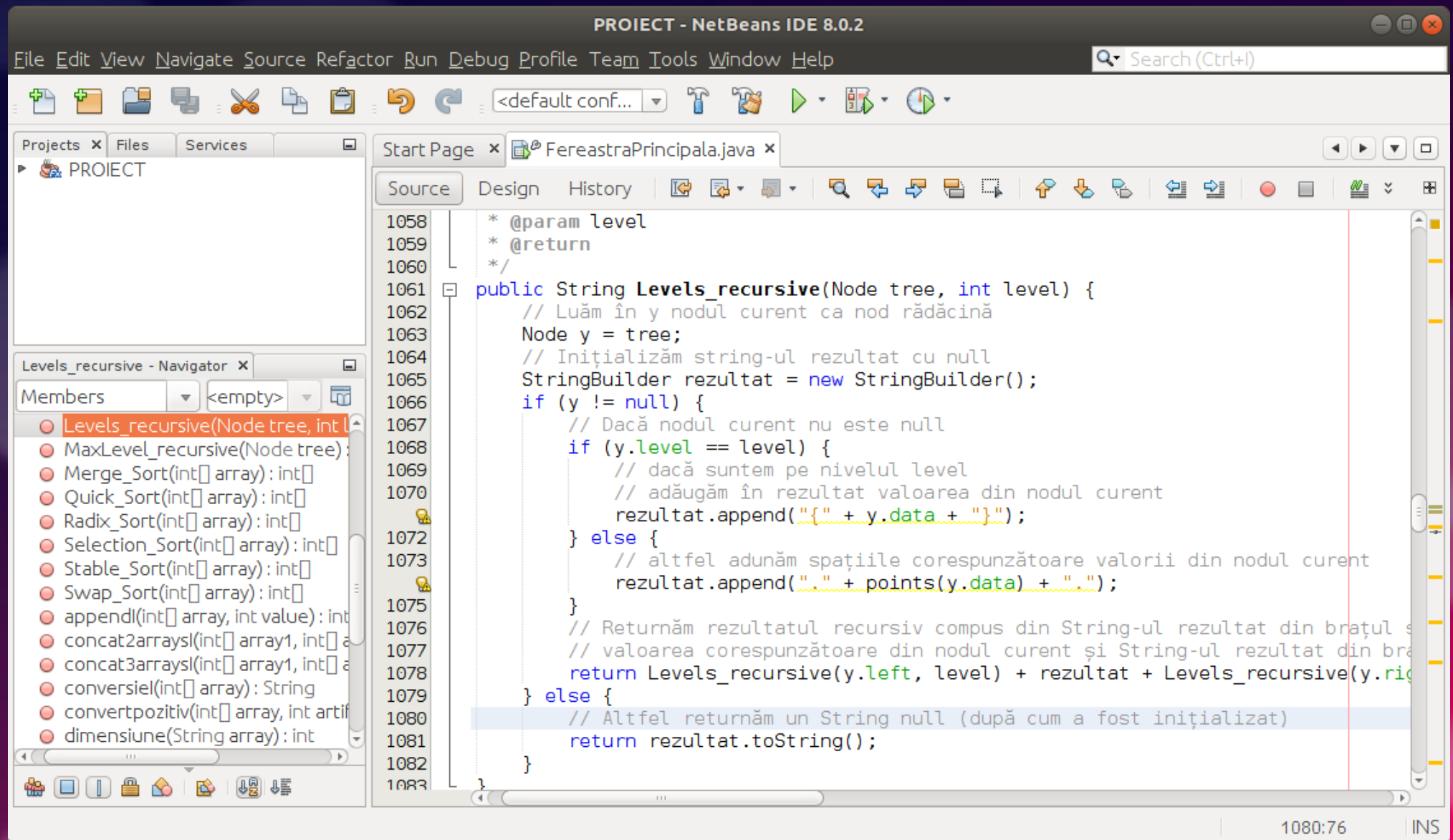
5. Programare Java

f. Puncte în locul numerelor



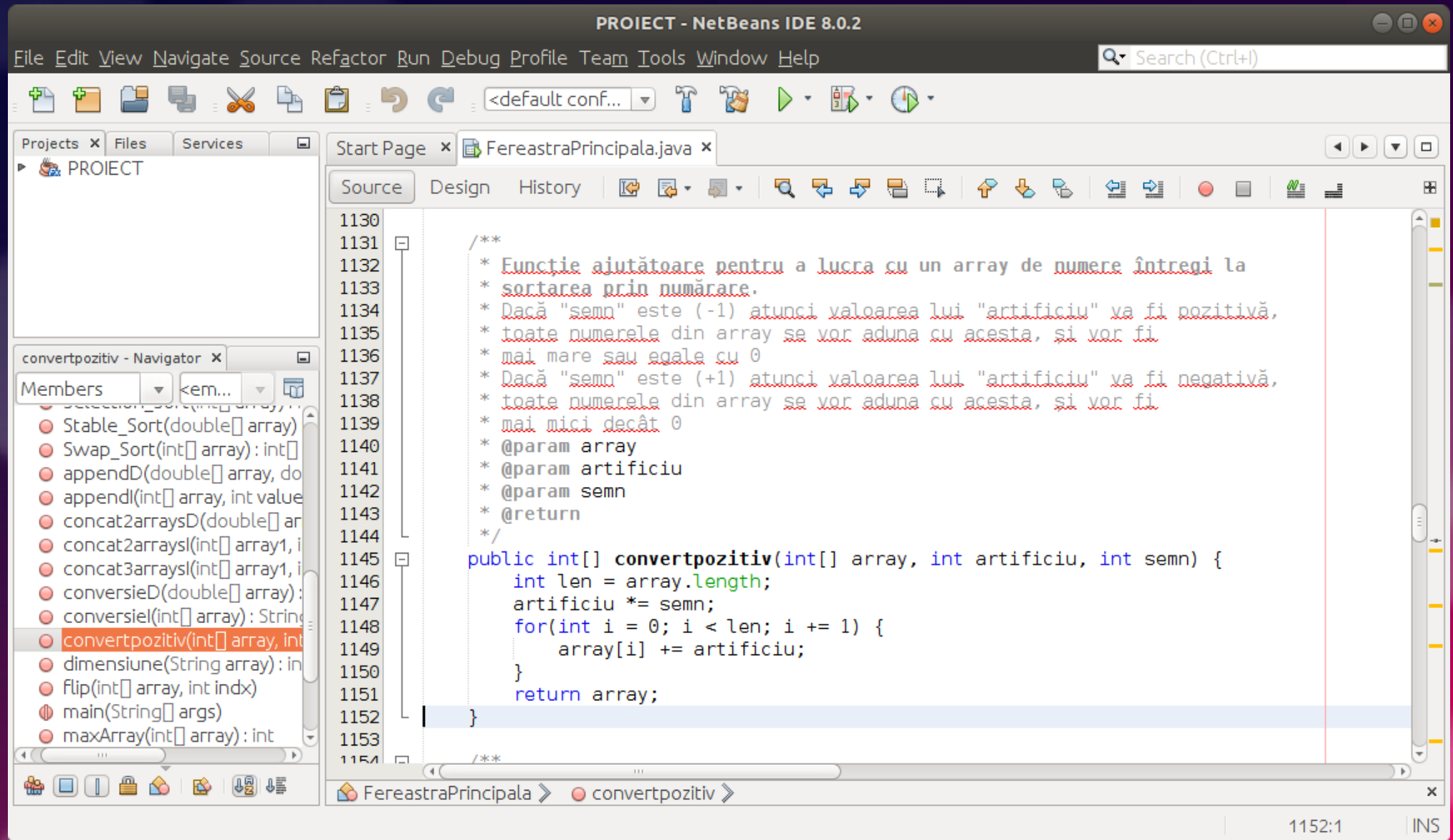
5. Programare Java

g. Vizualizare grafică a arborelui



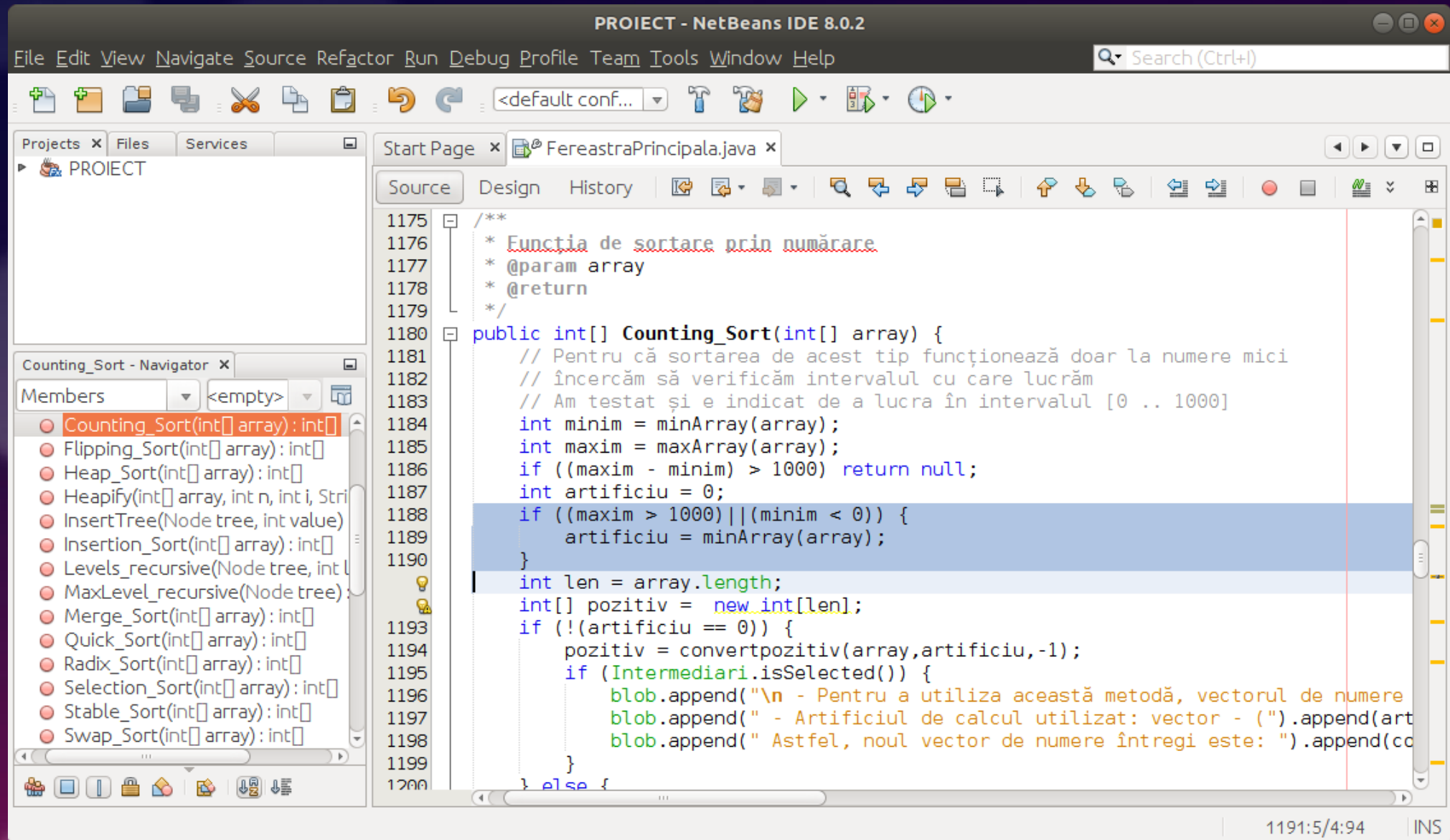
5. Programare Java

h. Artificiu de calcul



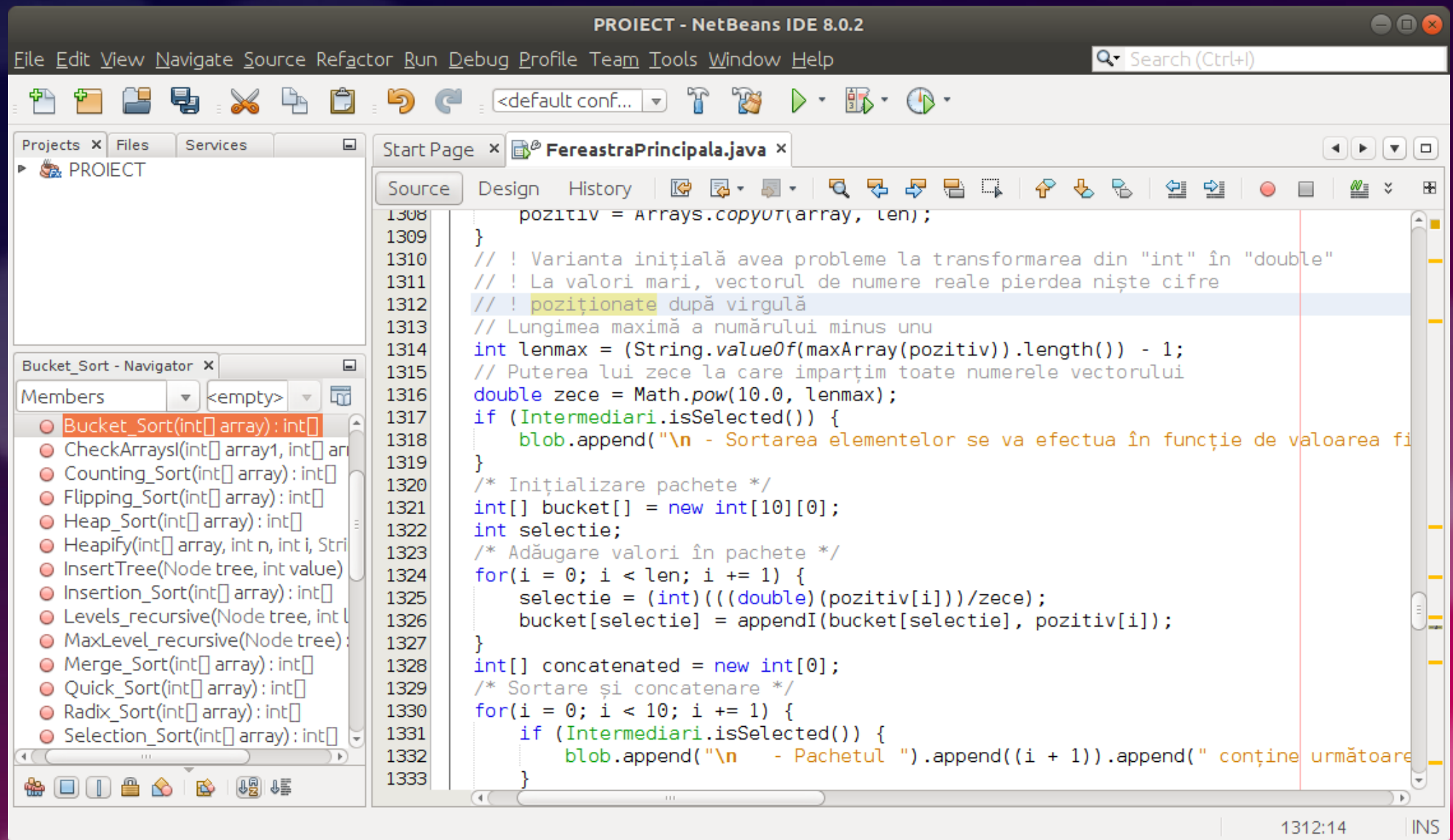
5. Programare Java

i. Restricții speciale



5. Programare Java

j. Conversii și cifre



6. Testarea aplicației

a. Depanări

PROIECT - NetBeans IDE 8.0.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Start Page x FereastraPrincipala.java x

Source Design History

```
1218 }
1219 if (Intermediari.isSelected()) {
1220     blob.append("\n Vectorul C ajutor se modifică în baza val
1221 }
1222 /* Acum C[i] conține numărul de elemente mai mic sau egal cu i
1223 int[] B = new int[len];
1224 for(int i = (len - 1); i >= 0; i -= 1) {
1225     B[C[pozitiv[i]] - 1] = pozitiv[i];
1226     C[pozitiv[i]] -= 1;
1227     if (Intermediari.isSelected()) {
1228         blob.append("\n Luăm elementul ").append((i + 1)).append("
1229         blob.append("\n - vector[ ").append((i + 1)).append(" =
1230         blob.append("C[ ").append(pozitiv[i]).append(" = ").app
1231         blob.append("B[ ").append((C[pozitiv[i]] + 1)).append(")
```

Variables x

Name	Type
array	int[]
minim	int
maxim	int
artificiu	int
len	int
pozitiv	int[]
k	int

Debugger

Algoritmi de sortare

Introduceți un vector de numere întregi pentru

uhjwb-fna21 20,3.12n bj24 214 fsdf-24 -120rfvv

Complexitatea algoritmului selectat

	Nefavorabil	Mediu	Favorabil
Merge			
Quick			
Insertion	O(k+n)	O(k+n)	O(k+n)
Bubble			
Selection			
Swap			
Heap			
BST			
Counting			
Bucket			
Radix			
Flipping			
Arată pași			

Rezultatul sortării

uhjwb-fna21 20,3.12n bj24 214 fsdf-24 -120rfvv

PROIECT (jfxsa-debug) running... 1227:1 INS

6. Testarea aplicației

b. Rezultate cu explicații

Algoritmi de sortare

Introduceți un vector de numere întregi pentru sortare

uhjwb-fna21 20,3.12n bj24 214 fsdf-24 -120rfvv

☐ Merge

☐ Quick

☐ Insertion

☐ Bubble

☐ Selection

☐ Swap

☐ Heap

☒ BST

☐ Counting

☐ Bucket

☐ Radix

☐ Flipping

☒ Arată pași

Complexitatea algoritmului selectat

Nefavorabil

Mediu

Favorabil

$O(n^2)$

$O(n \cdot \log(n))$

$O(n \cdot \log(n))$

Execută sortarea

Rezultatul sortării

Numărul total de numere întregi detectate: 8.

Numerele detectate sunt: {21,20,3,12,24,214,-24,-120}.

Metoda utilizată: 'Sortarea cu arbori de căutare binari'

Inserăm valorile vectorului într-un arbore cu radacina {21}

- Introducem {20} sub nodul {21} la stânga
- Introducem {3} sub nodul {20} la stânga
- Introducem {12} sub nodul {3} la dreapta
- Introducem {24} sub nodul {21} la dreapta
- Introducem {214} sub nodul {24} la dreapta
- Introducem {-24} sub nodul {3} la stânga
- Introducem {-120} sub nodul {-24} la stânga

Valorile de pe nivelul 0:{21}.....

Valorile de pe nivelul 1:{20}....{24}.....

Valorile de pe nivelul 2:{3}.....{214}

7. Concluzii

Scopul proiectului este unul didactic.

Am aplicat principiile teoretice pentru a vedea funcționarea lor în practică.

Concret, am dezasamblat bucătică cu bucătică un „mecanism” pentru a vedea cum funcționează și din ce este compus.