**Microlecție**

**Profesor:** Bledea Mihaela Alexandra

**Data:** 23.03.2023

Clasa: a X-a

**Profil/Specializare:** Real/Matematica-Informatica,   intensiv informatică

**Disciplina:** Informatică

**Unitatea de învățare:** Subprograme recursive

**Lecţia:** Recursivitatea pe divizori

**Tipul lecţiei:** Lecţia de fixare şi formare de priceperi şi deprinderi

**Competențe generale:**

1.   Identificarea conexiunilor dintre informatică şi societate.

2.   Identificarea datelor care intervin într-o problemă şi a relaţiilor dintre acestea.

3.   Elaborarea algoritmilor de rezolvare a problemelor.

4.   Aplicarea algoritmilor fundamentali în prelucrarea datelor.

5.   Implementarea algoritmilor într-un limbaj de programare.

**Competențe specifice:**

1. Utilizarea corectă a subprogramelor predefinite şi a celor definite de utilizator.

2. Construirea unor subprograme pentru rezolvarea subproblemelor unei probleme.

3. Aplicarea mecanismului recursivităţii prin crearea unor subprograme recursive (definite de utilizator).

4. Compararea dintre implementarea recursivă şi cea iterativă a aceluiaşi algoritm.

**Obiective operaţionale:**

1. Să construiască subprograme recursive pentru rezolvarea unei probleme.
2. Să descrie metoda de rezolvare a unei probleme în termeni recursivi
3. Să prezinte etapele de elaborare ale unui subprogram recursiv
4. Să definească condiția de oprire a unui program recursiv

**Desfășurarea lecției:**

|  |  |  |
| --- | --- | --- |
| **Evenimentele lecţiei** | **Activitatea din lecţie** | **Strategia didactică și evaluarea**  (metode, mijloace, forme de organizare a activităţii, metode de evaluare) |
| **Moment organizatoric** | * Pregătirea materialului didactic necesar desfășurării lecției. * Se asigură că fiecare elev are access la calculator și poate participa active la lecție | Conversația  Verificarea |
| **Captarea atenției** | * O scurtă poveste introductivă, însoțită de un exercițiu de imaginație, din care elevii înțeleg conceptul de recursivitate. * Să ne punem în situaţia pescarului din poveste care pescuieşte un peşte care îndeplineşte trei dorinţe. Ce aţi cere peştişorului? * Dacă niciun elev nu răspunde cum ne așteptăm, vom spune noi: * Cum ar fi dacă am cere peștișorului încă trei dorințe? | Povestire  Dialog |
| **Reactualizarea cunoștințelor însușite anterior** | * + Profesorul realizează reactualizarea cunoștințelor printr-un set de întrebări: * Ce reprezintă recursivitatea? * Răspuns: Prin recursivitate se înţelege reluarea repetitivă a unor paşi ai unui algoritm. * Cum se oprește un program recursiv? * Răspuns: Un program recursive se oprește cu ajutorul condiției de oprire. * Care este scopul condiției de oprire? De ce este importantă? * Răspuns: Fără o condiție de oprire, se ajunge la un ciclu infinit, iar resursele sunt limitate. | Conversaţia |
| **Enunțarea obiectivelor** | * + Anunţarea subiectului temei respective;   + Anunţarea obiectivelor urmărite;   + Anunţarea modului de desfăşurare a activităţii | Conversația |
| **Dirijarea învățării** | * Profesorul prezintă problemele care se vor rezolva. * La fiecare problemă, profesorul întreabă elevii cum s-ar rezolva problema. * Elevii sunt lasați să se gândească câteva minute. * Dacă niciunul nu vine cu vreo idee, se începe rezolvarea problemei împreună.  1. Să se scrie un subprogram recursiv, în care se citește, de la tastatură, un număr natural n. Să se calculeze și să se afișeze suma divizorilor săi.   #include <iostream> #include <math.h>  using namespace std;  int suma\_recursiva\_divizori(int numar, int divizor){   if(divizor >= 1){  if(numar % divizor == 0){  return divizor + suma\_recursiva\_divizori(numar, divizor - 1);  }  else {  return suma\_recursiva\_divizori(numar, divizor - 1);  }  }   return 0; }  int main() {  int numar;   cout << "Dati un numar natural: ";  cin >> numar;   cout << "Suma divizorilor numarului " << numar << " este: " << suma\_recursiva\_divizori(numar,numar);   return 0; }   1. Să se scrie un subprogram recursiv, în care se citește, de la tastatură, un număr natural n. Să se verifice și să se afișeze dacă numărul este prim.   #include <iostream> #include <math.h> using namespace std;  bool verificareNumarPrim(int numar, int i) {  if (numar <= 1)  return false;   if (numar == 2)  return true;   if (numar % i == 0)  return false;   if (i > sqrt(numar))  return true;   return verificareNumarPrim(numar, i + 1); }  int main() {  int numar;  int i = 2;   cout << "Introduceti un numar natural: ";  cin>> numar;   if (verificareNumarPrim(numar, i))  cout << "Numarul " << numar << " este prim.";  else  cout << "Numarul " << numar << " nu este prim.";   return 0; }   1. Să se scrie un subprogram recursiv, în care se citește, de la tastatură, un număr natural n. Să se verifice dacă este număr perfect.   // nr perfect #include <iostream> #include <math.h>  using namespace std;  int suma\_recursiva\_divizori(int numar, int divizor){   if(divizor >= 1){  if(numar % divizor == 0 && divizor != numar){  return divizor + suma\_recursiva\_divizori(numar, divizor - 1);  }  else {  return suma\_recursiva\_divizori(numar, divizor - 1);  }  }   return 0; }  int main() {   int numar, suma;   cout << "Dati un numar natural: ";  cin >> numar;   suma = suma\_recursiva\_divizori(numar,numar);   if(suma == numar)  {  cout << "Numarul " << numar << " este perfect";  }  else  {  cout << "Numarul " << numar << " nu este perfect";  }   return 0; } | Conversația  Exercițiu practic pe calculator |
| **Obținerea de performanță** | * Profesorul propune spre rezolvare, următoarele exerciții. Exercițiile care rămân nefăcute, vor fi ca tema de casă pentru data viitoare.      1. Să se scrie un subprogram recursiv, în care se citește, de la tastatură, un număr natural n. Să se calculeze și să se afișeze, divizorii săi, precum și numărul acestora.   #include <iostream> #include <iostream> #include <math.h>  using namespace std;  int numar\_divizori\_recursiv(int numar, int divizor){   if(divizor >= 1){  if(numar % divizor == 0){  cout<< divizor << " este divizor " << endl;  return 1 + numar\_divizori\_recursiv(numar, divizor - 1);  }  else {  return numar\_divizori\_recursiv(numar, divizor - 1);  }  }   return 0; }  int main() {  int numar, result;  cout << "Dati un numar natural: "; cin >> numar;  result = numar\_divizori\_recursiv(numar,numar); cout << "Numarul de divizori ai lui " << numar << " este: " << result;  return 0;  }   1. Să se scrie un subprogram recursiv, în care se citește, de la tastatură, un număr natural n. Să se calculeze și să se afișeze suma divizorilor săi pari.   #include <iostream> #include <math.h>  using namespace std;  int suma\_recursiva\_divizori\_pari(int numar, int divizor){   if(divizor >= 1){  if(numar % divizor == 0){  return divizor + suma\_recursiva\_divizori\_pari(numar, divizor - 2);  }  else {  return suma\_recursiva\_divizori\_pari(numar, divizor - 2);  }  }   return 0; }  int main() {  int numar;   cout << "Dati un numar natural: ";  cin >> numar;    if(numar % 2 == 0){  cout << "Suma divizorilor pari a numarului " << numar << " este " << suma\_recursiva\_divizori\_pari(numar,numar);  }  else {  cout << "Suma divizorilor pari a numarului " << numar << " este 0";  }  return 0; } | Conversația  Exercțiu practic pe calculator |
| **Asigurarea feedback-ului** | Profesorul face aprecieri asupra modului în care s-a desfășurat activitatea. | Conversația |