**Tema 28** Tehnica Reluării (Backtracking)

**Ex1.**

|  |  |
| --- | --- |
| **Descrierea exercitiului** | Privește video de la lecția dată despre roboțel și labirint. Codul de mai jos reprezintă o porțiune a programului care ar găsi ieșirea din labirint datorită metodei reluării. Substituie comentariul cu codul necesar astfel, încît acest program să fie funcționabil și roboțelul să găsească ieșirea corect. |
| **Ce trebuie sa fie initial în caseta compilatorului?** | #include <stdio.h>  #include <stdlib.h>  int a[100][100] = {  {-1, -1, -1, -1, -1, -1, -1 },  {-1, -1, -1, 0, -1, -1, -1 },  {-1, 0, 0, 0, 0, 0, -1 },  {-1, 0, -1, -1, -1, 0, -1 },  {-1, 0, 0, 0, -1, 0, -1 },  {-1, 0, -1, 0, 0, 0, 0 },  {-1, 0, -1, -1, -1, -1, -1 }  };  int n, m, xin, yin, xout, yout;  void miscare(int x, int y, int d)  {  if (x < 0 || x >= n)  return;  if (y < 0 || y >= m )  return;  if (a[x][y] != 0)  return;  a[x][y] = d;  if (x == xout && y == yout)  {  if (a[5][6] != 0)  printf("Corect ! Iesirea are ponderea %d", a[5][6]);  exit(-1);  }  else  {  /\*  aici lipsesc apelurile recursive  \*/  }  a[x][y] = 0;  }  int main()  {  n = m = 7;  xin = 6;  yin = 1;  xout = 5;  yout = 6;  miscare(xin, yin, 1);  return 0;  } |
| Ce trebuie să facă elevul? | **Să substituie**    /\*  aici lipsesc apelurile recursive  \*/  **Cu codul corect.**  **Codul correct:**  #include <stdio.h>  #include <stdlib.h>  int a[100][100] = {  {-1, -1, -1, -1, -1, -1, -1 },  {-1, -1, -1, 0, -1, -1, -1 },  {-1, 0, 0, 0, 0, 0, -1 },  {-1, 0, -1, -1, -1, 0, -1 },  {-1, 0, 0, 0, -1, 0, -1 },  {-1, 0, -1, 0, 0, 0, 0 },  {-1, 0, -1, -1, -1, -1, -1 }  };  int n, m, xin, yin, xout, yout;  void miscare(int x, int y, int d)  {  if (x < 0 || x >= n)  return;  if (y < 0 || y >= m )  return;  if (a[x][y] != 0)  return;  a[x][y] = d;  if (x == xout && y == yout)  {  if (a[5][6] != 0)  printf("Corect ! Iesirea are ponderea %d", a[5][6]);  exit(-1);  }  else  {  miscare(x, y + 1, d + 1);  miscare(x - 1, y, d + 1);  miscare(x, y - 1, d + 1);  miscare(x + 1, y, d + 1);  }  a[x][y] = 0;  }  int main()  {  n = m = 7;  xin = 6;  yin = 1;  xout = 5;  yout = 6;  miscare(xin, yin, 1);  return 0;  } |
| **Care este rezultatul corect afisat de compilator dupa ce elevul introduce codul correct?** | Corect ! Iesirea are ponderea 9 |

**Ex2.**

|  |  |
| --- | --- |
| **Descrierea exercitiului** | Pornește și analizează codul de mai jos. Este un program care generează și afișează toate posibilitățile de a descompune un număr n. |
| **Ce trebuie sa fie initial în caseta compilatorului?** | #include <stdio.h>  int z[50], n;  void afisare(int limita)  {  for (int i = 0; i <= limita - 1; i++)  printf("%d + ", z[i]);  printf("%d<br> ", z[limita]);  }  void descompunere(int n, int k)  {  if (n == 0)  afisare(k - 1); // afisare solutie si revenire  else  for (int i = 1; i <= n; i++) // pasi inainte  {  z[k] = i;  descompunere(n - i, k + 1);  }  }  int main()  {  n = 3;  descompunere(n, 0);  return 0;  } |
| Ce trebuie să facă elevul? | Să pornească și să analizeze programul. Este un exercițiu de analiză. |
| **Care este rezultatul corect afisat de compilator dupa ce elevul introduce codul correct?** | 1 + 1 + 1 1 + 2 2 + 1 3 |

Sarcini practice:

1. Cum crezi există o diferență dintre tehnica Forța Brută și Metoda Reluării ? Dacă da, atunci care este ea ?
2. Încearcă să cauți pe internet informație adăugătoare despre problema celor N regine. Această problemă poate fi rezolvată cu ajutorul Metodei Reluării. Un program ce te poate ajuta se află la acest link:

<https://drive.google.com/file/d/1a7EfEL-Ocee5cSYCGMOXOwcAR0_2dEz-/view?usp=sharing>

Analizează acest program și încearcă să îl modific astfel încît să se afișeze doar prima soluție găsită. (Încearcă să folosești funcția exit(0) ).

29. Grafuri. Noțiuni. Reprezentări

EX1.

|  |  |
| --- | --- |
| **Descrierea exercitiului** | În imaginea dată este reprezenat un graf neorientat. Iar reprezentarea lui sub formă de matrice de adiacență este prezentă în cod.  Modifică programul astfel încît graful final să fie acesta. (La ecran sunt afișate toate arcele din acest graf). |
| **Ce trebuie sa fie initial în caseta compilatorului?** | #include <stdio.h>  int n = 7;  int a[100][100] = {  {0, 0, 0, 0, 1, 1, 1},  {0, 0, 0, 1, 0, 1, 1},  {0, 0, 0, 1, 0, 1, 0},  {0, 1, 1, 0, 1, 1, 0},  {1, 0, 0, 1, 0, 0, 1},  {1, 1, 1, 1, 0, 0, 0},  {1, 1, 0, 0, 1, 0, 0}  };  void afisare\_lista()  {  int i, j;  printf("\n");  for (i = 0; i < n; i++)  {  for (j = i; j < n; j++)  if (a[i][j] == 1)  printf("(%d, %d) ", i + 1, j + 1);  }  }  int main()  {  afisare\_lista();  return 0;  } |
| Ce trebuie să facă elevul? | Să modifice codul și să îl aducă la forma data.  #include <stdio.h>  int n = 7;  int a[100][100] = {  {0, 0, 0, 0, 0, 0, 1},  {0, 0, 0, 1, 0, 1, 0},  {0, 0, 0, 1, 0, 1, 0},  {0, 1, 1, 0, 1, 1, 1},  {0, 0, 0, 1, 0, 0, 1},  {0, 1, 1, 1, 0, 0, 0},  {1, 0, 0, 1, 1, 0, 0}  };  void afisare\_lista()  {  int i, j;  printf("\n");  for (i = 0; i < n; i++)  {  for (j = i; j < n; j++)  if (a[i][j] == 1)  printf("(%d, %d) ", i + 1, j + 1);  }  }  int main()  {  afisare\_lista();  return 0;  } |
| **Care este rezultatul corect afisat de compilator dupa ce elevul introduce codul correct?** | (1, 7) (2, 4) (2, 6) (3, 4) (3, 6) (4, 5) (4, 6) (4, 7) (5, 7) |

Ex2

|  |  |
| --- | --- |
| **Descrierea exercitiului** | În imaginea dată este reprezenat un graf neorientat ponderat.  Modifică programul astfel încît la ecran să fie afișat corect acest graf împreună cu ponderile. Un arc se afișează altfel:  (sursa, destinatia, ponderea).  Exemplu: (1, 6, 1) (1, 5, 6) (1, 7, 7) (2, 7, 4) … etc. |
| **Ce trebuie sa fie initial în caseta compilatorului?** | #include <stdio.h>  int n = 6;  int a[100][100] = {  {0, 0, 0, 0, 0, 0},  {0, 0, 0, 1, 0, 1},  {0, 0, 0, 1, 0, 1},  {0, 1, 1, 0, 1, 1},  {0, 0, 0, 1, 0, 0},  {0, 1, 1, 1, 0, 0},  {1, 0, 0, 1, 1, 0}  };  void afisare\_lista()  {  int i, j;  printf("\n");  for (i = 0; i < n; i++)  {  for (j = i; j < n; j++)  if (a[i][j] == 1)  printf("(%d, %d) ", i + 1, j + 1);  }  }  int main()  {  afisare\_lista();  return 0;  } |
| Ce trebuie să facă elevul? | Să modifice codul și să îl aducă la forma data.  #include <stdio.h>  int n = 7;  int a[100][100] = {  {0, 0, 0, 0, 6, 1, 7},  {0, 0, 0, 0, 0, 0, 4},  {0, 0, 0, 3, 0, 2, 0},  {0, 0, 3, 0, 5, 0, 0},  {6, 0, 0, 5, 0, 0, 0},  {1, 0, 2, 0, 0, 0, 0},  {7, 0, 0, 0, 0, 0, 0}  };  void afisare\_lista()  {  int i, j;  printf("\n");  for (i = 0; i < n; i++)  {  for (j = i; j < n; j++)  if (a[i][j] != 0)  printf("(%d, %d, %d) ", i + 1, j + 1, a[i][j]);  }  }  int main()  {  afisare\_lista();  return 0;  } |
| **Care este rezultatul corect afisat de compilator dupa ce elevul introduce codul correct?** | (1, 5, 6) (1, 6, 1) (1, 7, 7) (2, 7, 4) (3, 4, 3) (3, 6, 2) (4, 5, 5) |

Sarcini practice:

1. Realizează un program care va citi de la tastatură o matrice de adiacență pentru un graf orientat. Atrage atenția la faptul că graful e orientat. Apoi afișează rezultatul sub formă de listă și matrice de adiacență.
2. Încearcă să faci un program care va citi de la tastatură sub formă de listă n arce a unui graf neorientat ponderat iar apoi va afișa la ecran rezultatul sub formă de matrice de adiacență.