**Tema 22**

**Ex2**

|  |  |
| --- | --- |
| **Descrierea exercitiului** | Programul dat va sorta un șir de numere în ordine crescătoare cu ajutorul algoritmului bubble sort. Modifică codul astfel încît elementele să fie sortate în ordine descrăscătoare. |
| **Ce trebuie sa fie initial în caseta compilatorului?** | #include <stdio.h>  void afisaretablou(int x[], int n)  {  for (int i = 0; i < n; i++)  printf("%d ", x[i]);  return;  }  void sortare(int x[], int n)  {  int e\_permutat;  do  {  e\_permutat = 0;  for (int i = 0; i < n - 1; i++)  if (x[i] > x[i + 1])  {  int tmp = x[i];  x[i] = x[i + 1];  x[ i + 1] = tmp;  e\_permutat = 1;  }  } while (e\_permutat == 1);  }  int main()  {  int a[] = { 101, 45, -23, 0, -5, 77, 44, 3, 2 };  int n = sizeof(a) / sizeof(a[0]);  sortare(a, n);  afisaretablou(a, n);  } |
| **Ce trebuie să facă elevul?** | **Să modifice linia dată de cod**  if (x[i] > x[i + 1])  **Să înlocuiască cu**  if (x[i] < x[i + 1])  **Cod corect:**  #include <stdio.h>  void afisaretablou(int x[], int n)  {  for (int i = 0; i < n; i++)  printf("%d ", x[i]);  return;  }  void sortare(int x[], int n)  {  int e\_permutat;  do  {  e\_permutat = 0;  for (int i = 0; i < n - 1; i++)  if (x[i] > x[i + 1])  {  int tmp = x[i];  x[i] = x[i + 1];  x[ i + 1] = tmp;  e\_permutat = 1;  }  } while (e\_permutat == 1);  }  int main()  {  int a[] = { 101, 45, -23, 0, -5, 77, 44, 3, 2 };  int n = sizeof(a) / sizeof(a[0]);  sortare(a, n);  afisaretablou(a, n);  } |
| **Care este rezultatul corect afisat de compilator dupa ce elevul introduce codul correct?** | 101 77 45 44 3 2 0 -5 -23 |

**Tema 23**

**Ex1**

|  |  |
| --- | --- |
| **Descrierea exercitiului** | După cum ați aflat, algoritmul bubble sort este destul de lent. Din acest motiv trebuie să cunoaștem și alți algoritmi de sortare un pic mai eficienți. În funcția selection\_sort() lipsește implementarea algoritmului. Scrie codul lipsă și pornește programul. Funcția main nu trebuie să fie modificată.  Salveaza acest program deoarece la pasul urmator ai sa ai nevoie de el ! |
| **Ce trebuie sa fie initial în caseta compilatorului?** | #include <stdio.h>  void print(int a[], int n)  {  for (int i = 0; i < n; i++)  printf("%d ", a[i]);  }  void selection\_sort(int a[], int n)  {  /\* scrie implementarea aici \*/  }  int main()  {  int a[] = {49, 32, 15, -5, 3, -5, 101};  int n = sizeof(a) / sizeof(a[0]);  selection\_sort(a, n);  print(a, n);  } |
| **Ce trebuie să facă elevul?** | **Să modifice linia dată de cod**  /\* scrie implementarea aici \*/  **Să înlocuiască cu**  int min\_index, aux;  for (int i = 0; i < n; i++)  {  min\_index = i;  for (int j = i + 1; j < n; j++)  if (a[j] < a[min\_index])  min\_index = j;  aux = a[min\_index];  a[min\_index] = a[i];  a[i] = aux;  }  **Cod Corect**  #include <stdio.h>  void print(int a[], int n)  {  for (int i = 0; i < n; i++)  printf("%d ", a[i]);  }  void selection\_sort(int a[], int n)  {  int min\_index, aux;  for (int i = 0; i < n; i++)  {  min\_index = i;  for (int j = i + 1; j < n; j++)  if (a[j] < a[min\_index])  min\_index = j;  aux = a[min\_index];  a[min\_index] = a[i];  a[i] = aux;  }  }  int main()  {  int a[] = {49, 32, 15, -5, 3, -5, 101};  int n = sizeof(a) / sizeof(a[0]);  selection\_sort(a, n);  print(a, n);  return 0;  } |
| **Care este rezultatul corect afisat de compilator dupa ce elevul introduce codul correct?** | -5 -5 3 15 32 49 101 |

**Ex2**

|  |  |
| --- | --- |
| **Descrierea exercitiului** | Modifică exercițiul de la exemplul trecut astfel încît selection sort să sorteze elementele în ordine descrescătoare. Funcția main râmîne neatinsă. |
| **Ce trebuie sa fie initial în caseta compilatorului?** | #include <stdio.h>  void print(int a[], int n)  {  for (int i = 0; i < n; i++)  printf("%d ", a[i]);  }  void selection\_sort(int a[], int n)  {  /\* scrie implementarea aici \*/  }  int main()  {  int a[] = {49, 32, 15, -5, 3, -5, 101};  int n = sizeof(a) / sizeof(a[0]);  selection\_sort(a, n);  print(a, n);  } |
| **Ce trebuie să facă elevul?** | **Să modifice linia dată de cod**  /\* scrie implementarea aici \*/  **Să înlocuiască cu**  int max\_index, aux;  for (int i = 0; i < n; i++)  {  max\_index = i;  for (int j = i + 1; j < n; j++)  if (a[j] > a[max\_index])  max\_index = j;  aux = a[max\_index];  a[max\_index] = a[i];  a[i] = aux;  }  **Cod Corect**  #include <stdio.h>  void print(int a[], int n)  {  for (int i = 0; i < n; i++)  printf("%d ", a[i]);  }  void selection\_sort(int a[], int n)  {  int max\_index, aux;  for (int i = 0; i < n; i++)  {  max\_index = i;  for (int j = i + 1; j < n; j++)  if (a[j] > a[max\_index])  max\_index = j;  aux = a[max\_index];  a[max\_index] = a[i];  a[i] = aux;  }  }  int main()  {  int a[] = {49, 32, 15, -5, 3, -5, 101};  int n = sizeof(a) / sizeof(a[0]);  selection\_sort(a, n);  print(a, n);  return 0;  } |
| **Care este rezultatul corect afisat de compilator dupa ce elevul introduce codul correct?** | 101 49 32 15 3 -5 -5 |

**Tema 28.2**

**La sfirsit de raspuns trebuie sa fie un spatiu (dupa <br>)**

**Tema 29**

|  |  |
| --- | --- |
| **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;  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;  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) |

**Ex 2**

|  |  |
| --- | --- |
| **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;  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;  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) |