**Tema 22**

**Ex1**

|  |  |
| --- | --- |
| **Descrierea exercitiului** | Codul din căsuță este greșit. El reprezintă algoritmul de sortate Bubble Sort. Corectează codul astfel încît programul să funcționeze și să sorteze corect tabloul. |
| **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)  {  do  {  e\_permutat = 1;  for (int i = 0; i < n; i++)  if (x[i] > x[i + 1])  {  int tmp = x[i]  x[i] = x[i + 1];  x[i + 1] = tmp;  e\_permutat = 0  }  } 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 corecteze greșelile în cod.**  **Codul 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?** | -23 -5 0 2 3 44 45 77 101 |

**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]);  printf("\n");  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]) |
| **Care este rezultatul corect afisat de compilator dupa ce elevul introduce codul correct?** | 101 77 45 44 3 2 0 -5 -23 |

**Sarcini practice:**

1. Rescrie algoritmul bubble sort fără a utiliza fanionul (variabila e\_permutat). Încearcă să cauți careva variante de implementare și în surse externe. Analizează codul.
2. Încearcă să explici unui coleg principiul de lucru a acestui algoritm (fără cod, doar algoritmul).

**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]);  printf("\n");  }  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]);  printf("\n");  }  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]);  printf("\n");  }  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]);  printf("\n");  }  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 |

**Sarcini practice:**

1. Cum crezi de ce totuși algoritmul Selection Sort este mai rapid ca Bubble Sort. Oare Selection Sort face mai puține permutări în tablou ca Bubble Sort ?
2. Modifică algoritmul Selection Sort astfel încît el să lucreze pentru sortarea caracterelor. Simulează sortarea cu un șir de caractere. Nu uita să afișezi corect caracterele pe ecran. (Nu uita că caracterele sunt ordonate conform codurilor din ASCII).