Rossi Vincenzo matricola 12/345

Algoritmo di Insertion sort

Implementazione dell'algoritmo Insertion Sort come funzione:

```
void insertion(int interi[100],int tot)
{
   int temp; /* Indice temporaneo per scambiare elementi */
   int prossimo;
   int attuale;
   for (prossimo=1;prossimo<=tot;prossimo++)
     {
      temp=interi[prossimo];
      attuale=prossimo;
      while ((attuale>0) && (interi[attuale-1]>temp))
      {
          interi[attuale]=interi[attuale-1];
          attuale=attuale-1;
       }
   interi[attuale]=temp;
   }
}
```

Scopo

Ordinamento di un array di numeri interi dato in ingresso

Specifiche

Versione per array di interi
Void insertionsort(int interi[100], int tot);

Descrizione

L'insertion sort è il più intuitivo dei tre algoritmi di ordinamento, è simile al modo che si potrebbe usare per ordinare un mazzo di carte. Supponendo che si parta con un mazzo già parzialmente ordinato, si prende la prossima carta e la si inserisce nel posto giusto tra le altre. Inserita questa, si passa alla successiva che viene inserita nel posto giusto. Dopo che tutte le carte sono state correttamente inserite, il mazzo di carte risulterà ordinato.

Indicatori di Errore

Nessuno

Indicazioni sull'utilizzo

Nessuna

Complessità di Tempo e Di Spazio

Complessità di tempo:

La complessità di tempo dipende dal numero di confronti effettuati nella fase di inizializzazione(n-1), e da quelli effettuati durante l'ordinamento vero e proprio(n(n-1)/2), quindi la complessità asintotica di tempo è: $(n^2 + n - 4)/2 = O(n^2)$

Complessità di spazio:

La struttura dati utilizzata per implementare l'algoritmo è un ARRAY monodimensionale, contenente i valori da ordinare, di conseguenza la complessità di spazio è O (n).

Esempio di Utilizzo

Esempio di programma che chiama le funzioni:

```
# include <stdio.h>
# define max 100
int i,j; /* Indici di scorrimento dell'array */
void insertion(int interi[max],int tot);
main()
{
int interi[max]; /* Array che contiene i valori da ordinare */
int tot; /* Numero totale di elementi contenuti nell'array */
/* Inserimento elementi nell'array */
printf("\nQuanti elementi deve contenere l'array: ");
scanf("%d",&tot);
for (i=1;i<=tot;i++)
{
printf("\nInserire il %d° elemento: ",i);
scanf("%d",&interi[i]);
}
else if (choice=='2')
{
insertion(interi,tot);
/* Visualizzazione array ordinato */
printf("\nArray Ordinato:");
for (i=1;i<=tot;i++)
printf(" %d",interi[i]);
printf("\n");
}
```

Esempi di esecuzione:
Dati in ingresso i numeri: 8 35 18 23
Con uno qualsiasi dei tre algoritmi si ottiene i uscita: 8 18 23 35
Dati in ingresso i numeri 87 90 63 86 72
Con uno qualsiasi dei tre algoritmi si ottiene i uscita: 63 72 86 87 90

Riferimenti Bibliografici

Introduction to Algorithms – Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest