**Analisi Requisiti**

* Inserimento degli elementi dell’array
* Controllo degli elementi pari e separazione dei dispari
* Ordinamento degli elementi pari
* Unione dei pari e dei dispari mantenendo invariate posizioni

**Analisi Funzionale**

* Per prima cosa faccio inserire all’utente un array di lunghezza minima 10 e massima di 11 elementi.
  + Durante l’inserimento controllo la presenza di doppioni e
  + Controllo che i numeri siano compresi tra 1 e 30
* Dopo l’inserimento del decimo richiedo all’utente se vuole che vengano inseriti ulteriori elementi
  + se la risposta è affermativa inserisco l’undicesimo e vado avanti
  + se la risposta è negativa interrompo l’inserimento.
* Utilizzo un metodo che controlla quali elementi dell’array sono pari
  + se l’elemento è pari lo copio in un secondo array
* Utilizzo un altro metodo che tramite un algoritmo di ordinamento mi ordini l’array di numeri pari in ordine crescente
* Tramite un metodo controllo quali elementi dell’array principale erano pari
  + se trovo un elemento pari nell’array iniziale allora copio il primo numero dell’array pari al suo posto e così via
  + in questo modo lascio invariata la posizione dei numeri dispari
* Stampo all’interno del main l’array ordinato

**Analisi Tecnica**

* Inserisco con un ciclo dieci elementi nell’array
  + Controllo che gli elementi inseriti non si ripetano e siano compresi tra 1 e 30 con un metodo *controllo*
* Richiedo se si vuole inserire anche l’undicesimo
* Terminato l’inserimento utilizzo il metodo *pari*
  + Controlla quali elementi sono pari e li copia in un array secondario
* Invoco il metodo *ordina* che come parametri utilizzerà l’array dei numeri pari
  + Ordinamento tramite quick sort
* Nel *main* copio gli elementi dell’array pari in quello originale senza variare le posizioni dei dispari