Data: 22/12/2020

Verifica di TPSIT, classe 4^BROB.

1. Scrivere una funzione **ricorsiva** chiamata `primo_Dispari` che data una Lista, che per valori ha numeri interi, restituisca il puntatore al primo elemento dispari nella lista (restituisce NULL se la lista è vuota o non contiene elementi dispari).

```
numero* primo_Dispari(numero* Node){

   if(((Node->valore)%2)== 1){
       return Node;
   }
   else{
        if(Node->next == NULL){
           printf("Non ci sono numeri dispari");
           return NULL;
       }
       primo_Dispari(Node->next);
   }

}
```

2. Scrivere una funzione iterativa, oppure ricorsiva (a tua scelta), `cancella`, che ricevuta una Lista e un intero n, elimini (**senza deallocare**) i primi n elementi e ritorni il puntatore alla testa della lista modificata .

```
numero* cancella(numero* Node, int n){

   for(int i = 0; i<n; i++){

       Node = Node->next;

   }

   return Node;

}
```

3. Data la struttura:

Data: 22/12/2020

```
typedef struct nodo
{
    int valore;
    struct nodo* successivo;
} Nodo;
```

e utilizzando la funzione (NON dovete implementarla):

```
Nodo* push(Nodo* lista, int x); //inserisce un elemento in fondo alla lista con il valore x e ritorna la head alla nuova lista
```

scrivere un programma che data una `Lista` di lunghezza arbitraria, crei una nuova lista che sia la copia esatta di `Lista`.

```c
int main(){
   //...

   for(int i = 0; i<lunghezzaLista_ricorsiva(lista1); i++){
       push(lista2, lista1->valore);
       lista1 = lista1->successivo;
   }
   //...
   return 0;
}

int lunghezzaLista_ricorsiva(numero* head){
   if(head == NULL){
       return 0;
   }
   else{
       return 1+lunghezzaLista_ricorsiva(head->next);
   }
}
```

4. Quali vantaggi ha una lista rispetto ad un array?

I vantaggi di una lista rispetto ad un array sono il numero di operazioni possibili che si possono fare. A differenza degli array le liste possono essere

manipolate molto più facilmente ed è possibile fare operazioni come l'eliminazione di un elemento, aggiungerne un altro, …