Settimana 3
Foglio di lavoro

1. Trasforma il programma dell'es 1 (analisi del DNA) in funzione.
Abbiamo gia' il corpo. Pensare con cura la definizione (header) della funzione contaBasi. Dato il solito array di char, contaBasi restituisce (come?) quante volte ciascuna base e' presente.

2. Scrivere una funzione cercaA che testa se un array di char contiene il carattere 'A'. La funzione restituisce (col return) true oppure false.

3. Riscrivere la funzione cercaA in modo che restituisca l'indice della prima occorrenza di 'A', e -1 altrimenti.
Modificare la funzione in modo che cerchi 'A' nel frammento di array che va dall'indice inizio all'indice fine.

4. Scrivi una funzione che cancella da un array l'elemento alla posizione i.
Qual e' l'idea? Di che parametri abbiamo bisogno? Quali sono i possibili problemi?

5. Scrivi una funzione che inserisce in un array un elemento x alla posizione i.
Qual e' l'idea? Di che parametri abbiamo bisogno? Quali sono i possibili problemi?

6. Ancora cercaA. Questa volta restituire un puntatore alla prima occorrenza di 'A'. E se 'A' non compare, come potremmo segnalarlo?

Esempi di sviluppo.

3. Cercare in un array.
La soluzione piu' elegante usa una sentinella per uscire dal ciclo:

```
//cerca il carattere ch nell'array A e restituisce l'indice
int cerca (char A[], int dim,  char ch){
        int i=0;
        bool trovato = false;
        //prima di i non ci sono occorrenze di ch
        //uscita: trovato e A[i] = ch, oppure non trovato e i=dim
        while (i<dim && !trovato)
                if (A[i]== ch)  trovato = true;
                else i=i+1;

        if (trovato) return i;
        else return -1;
}
```

Che ne pensiamo di quest'altra soluzione?

```
int cerca2 (char A[], int dim, char ch){
        int i=0;
        for (i=0;i<dim; i++ )
                if (A[i]== ch)  return i;
return -1;}
```

E' accettabile… ma brutale. L'uso di return (o break) all'interno di un ciclo e'

scoraggiato, in quanto non permette al corpo del ciclo di completare le operazioni previste (e' un po' come spegnere il computer staccando la spina o usando reset). In un ciclo un po' complesso, diventa difficile ragionare sulle condizioni di uscita dal ciclo (l'invariante), perche' l'uscita e' indipendente dalle condizioni sul ciclo.

4. Cancellare in un array.
Ci sono un paio di idee importanti.
Un array ha due lunghezze associate:

1. la dimensione di dichiarazione, che e' costante ed e' il numero di posti allocati in memoria per l'array. Questa dimensione e' fissa, e infatti deve essere una costante.
2. la lunghezza "logica", cioe' il numero di posti effettivamente usati: possiamo non riempire completamente l'array. Questa lunghezza varia se cancelliamo o inseriamo elementi.

Per cancellare l'elemento alla posizione i e mantenere l'array compatto (una sequenza contigua di dati) bisogna ***spostare gli elementi che seguono*** A[i], per prendere il posto dell'elemento cancellato.

```
//cancella l'elemento di indice posizione,
//spostando a sinistra gli elementi che seguono,
//e aggiorna la lunghezza logica (il numero di elementi utilizzati)

void cancella_a (char A[], int posizione, int & lung){

// sposta a sinistra gli elementi di indice da  (i+1) a (lung-1)
//lung-1 e' l'indice dell'ultimo elemento
for (int i=posizione; i<lung-1; i++ )
    A[i]=A[i+1];

lung= lung-1;  //aggiorniamo la dimensione logica
                //possibile grazie al passaggio per riferimento
}
```

Esempio. Questo array ha dimensione 6, ma la lunghezza della parte che utilizziamo e' 4. Quindi, ignoriamo gli elementi oltre il quarto (per es. in una stampa).

Dopo aver cancellato A, la parte utilizzata ha lunghezza 3.