Code

Code

- Una coda è un insieme ordinato, non limitato, di elementi dello stesso tipo su cui il primo elemento memorizzato e' anche il primo ad essere recuperato (FIFO=first input first output)

ADT coda

- Il tipo CODA è un ADT <S,F,C> dove

S = {coda, atomo, boolean}

- coda è il dominio di interesse, atomo è il dominio degli elementi che formano le liste

F = {incoda, front, null, outcoda}

- incoda : atomo x coda -> coda
  - nserisce un elemento alla fine della coda
- front : coda -> atomo
  - ritorna l'elemento in cima alla lista
- null : coda -> boolean
  - ritorna il valore vero se la coda è vuota
- outcoda : coda -> coda
  - ritorna la coda privata dell'elemento in cima

C = coda vuota,

- è la costante che denota la coda priva di elementi

Implementazione sequenziale

```
typedef int TAtomo;
typedef struct StCoda {
        int primo, ultimo, NMax;
        TAtomo * e;
        } Coda;

int InizializzaCoda( Coda *PC, int NElemMax )
{
    PC->e=(Tatomo*)malloc(sizeof(TAtomo)*NElemMax);
    if( PC->e == NULL ) return 0;;
    PC-> ultimo = 0;
    PC -> primo = -1;
    return PC->NMax = NElemMax;
}
```

Implementazione sequenziale

```
int null( Coda C ) {
        return (C.primo == -1);
}

int full( Coda C ) {
        return (C.primo == C.ultimo);
}

int outcoda ( Coda *PC ) {
        if( null(*PC) ) return 0;
        if( PC-> primo == PC ->NElemMax) PC->primo = 0
        else (PC->primo)++;
        if( PC-> ultimo == PC-> primo ) PC ->primo = -1
        return 0;
}
```

Implementazione sequenziale

```
TAtomo  front( Coda C ) {
        if( null(C) ) return 0;
        return C.e[C.primo];
}

int incoda (Coda *PC, TAtomo A ) {
        if( full(*PC)) return 0;
        if( null(*PC)
              { Pc -> ultimo =1;
                PC -> primo = 0;
        PC->e[PC->primo] = A
        }
      else  {
        PC->e[PC->ultimo] = A
        if (PC->ultimo == PC -> NelemMax - 1)
                PC->ultimo = 0
        else PC->ultimo ++)
      return 1;
}
```

Implementazione concatenata

```
typedef int TAtomo;
typedef struct Stelem {
        struct Selem *next;
        TAtomo info;
        } elem;
typedef struct StCoda {
        elem *primo, ,*ultimo;
        } TCoda, *PTCoda;

int null( TCoda C ) { return !C.primo;}
int outcoda ( PTCoda PC ) {
    elem *aux;
    if( null(*PC) ) return 0;
     aux = PC ->primo
     PC -> primo = PC ->primo ->next
     if (null(*PC))  PC ->ultimo = NULL
     free(aux);
     return 1;
}
```

```
TAtomo front( TCoda C ) {
        if( null(C) ) return NULL;
        return C.primo -> info;
}

int incoda( PTCoda PC, TAtomo A ) {
        elem *aux;
        aux = (elem *)malloc(sizeof(elem));
        if (!aux) return 0 ;
        aux ->info = A;
        aux -> next = NULL;
        if null(*PC) PC ->ultimo = PC->primo = aux;
        else {    PC ->ultimo -> next = aux;
              PC ->ultimo = aux
             }
        return 1;
}
```