

Scrivete IN STAMPATELLO nome, numero di matricola e data su tutti i fogli che consegnate, altrimenti non saranno presi in considerazione.

Domande (5 punti):

1. Descrivere cosa sono i parametri formali ed i parametri attuali di una chiamata a sottoprogramma. Fare un esempio
2. Cosa si intende per polimorfismo in un linguaggio di programmazione?
3. Descrivere le differenze tra procedura/funzione e macro
4. Cosa è un compilatore e che compiti ha? che differenze esistono con un interprete?
5. Per quale motivo i goto sono da evitare? Fare degli esempi

Esercizio 1 (3 punti) L'algoritmo **A** gira nel caso peggiore in un tempo pari a k^n , mentre l'algoritmo **B** gira nel caso peggiore in un tempo pari a $n!$, cosa possiamo affermare sui due algoritmi? Se **B** girasse nel caso peggiore in un tempo pari a $k \log n$ cosa si potrebbe dire?

NOTA di soluzione : k è da considerarsi costante si deve pensare ad n che tende all'infinito. L'ordine di grandezza del fattoriale è superiore all'esponenziale.

Esercizio 2 (3 punti) Considerando il linguaggio definito sull'alfabeto $\{a,b\}$ descrivere la macchina a stati che riconosce tutte le parole che iniziano per "aa"

NOTA: bastano due stati uno che dopo aver ricevuto inizialmente due a accetta tutto l'altro che non accetta.

Esercizio 3 (3 punti) Si consideri la seguente definizione in BNF di un linguaggio:

```
<tree> ::= <leaf> | "(" <children> ")"
<children> ::= <tree> | <tree> " " <children>
<leaf> ::= "a" | "b" | "c"
```

Commentare le seguenti espressioni evidenziando quale o quali fanno parte del linguaggio?

- $a \mid (b) \mid a$
- **$(a a c)$**
- $a (a c)$
- **$(a (c))$**

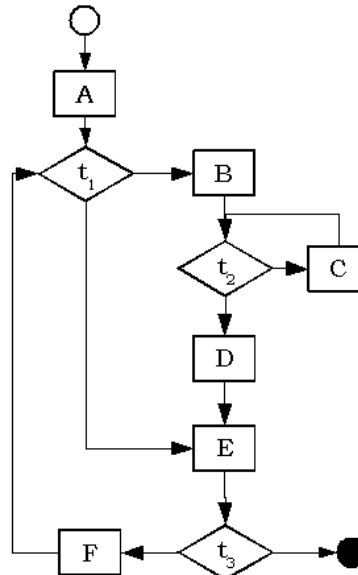
Esercizio 4 (4 punti) Si consideri il seguente programma in C dire cosa produce come output

```
int main(void)
{
    int c;
    while ((c = getchar()) != EOF) {
        if (c == ' ') {
            putchar(c);
            while((c = getchar()) == ' ' && c != EOF)
                ;
        }
        if (c == EOF)
            break;
        putchar(c);
    }
    return 0;
}
```

NOTA: si tratta di un programma che elimina gli spazi ripetuti da un flusso in ingresso

Scrivete IN STAMPATELLO nome, numero di matricola e data su tutti i fogli che consegnate, altrimenti non saranno presi in considerazione.

Esercizio 5 (5 punti) Si consideri il seguente programma non strutturato, dire per quale motivo non è strutturato e facendo le opportune supposizioni, presentarne una versione strutturata utilizzando la trasformazione di **Ashcroft Manna**. Commentare quanto ottenuto



NOTA: sistemare le frecce prima di iniziare a valutare i blocchi, la freccia entrante a sinistra del test t1 può essere messa sopra t1. B-t2-C-D-E-T1 non includono essi stessi elementi non strutturati ma in relazione a t3 e f abbiamo un ciclo che non è ne pre ne post condizionato

Esercizio 6 (4 punti) Consideriamo il problema di eliminare i doppi non consecutivi in un elenco di numeri (da 0 a 9) dati in ingresso.

Esempio: consideriamo in ingresso 11234426649 l'output dovrà essere 123469.

Facendo le opportune assunzioni scrivere uno pseudocodice che risolva il problema.

NOTA: Si può usare una struttura di appoggio per memorizzare la presenza dei doppi nella sequenza

Esercizio 7 (4 punti) Data la seguente funzione ricorsiva aiutandosi disegnando lo stack delle chiamate dire cose torna come risultato se chiamata passando come input n=5

```

int ciccia (int *n) {
    if (*n == 0) return 1;
    else return (*n * ciccia(*n - 1));
}
    
```

NOTA: c'è un problema di compatibilità di tipi dato che viene passato un tipo intero nella chiamata ricorsiva quando viene dichiarato un tipo puntatore a intero. Se fosse possibile convertire in maniera implicita quello che succederebbe è l'accesso a zone di memoria probabilmente illegali.