

Cognome: <b>Andrea Ferri</b>	Nome: <b>Ferri</b>
Matricola:	Anno immatricolazione:
Esame Totale 6 CFU Tempo a disposizione 1 ora	
<p>PROPEDEUTICITÀ: lo studente è consapevole che NON PUÒ SOSTENERE questo esame SE NON SI È SUPERATO l'esame di "Laboratorio di Programmazione di Sistema - LPS" (precedentemente "Laboratorio di Architettura degli Elaboratori - LAE")</p> <p>Lo studente dichiara di aver già superato l'esame di LPS (LAE).</p> <p>Lo studente dichiara di voler sostenere questo esame CON RISERVA SUL SUPERAMENTO DI LPS IN UNO DEGLI APPELLI DELLA SESSIONE CORRENTE. Lo studente è consapevole che PERDERÀ IL VOTO DI QUESTO ESAME SE NON SUPERERÀ LPS IN UNO DEGLI APPELLI DELLA SESSIONE CORRENTE.</p>	

- Domande a risposta multipla → 2 punti per ciascuna risposta esatta, -1 punti per ciascuna risposta errata, 0 punti per ogni risposta omessa. Le domande a risposta multipla possono avere una e una sola risposta esatta.
- Per indicare la risposta scelta, sottolinearla, oppure evidenziarla, oppure mettere una **X** sulla lettera, oppure riquadrare la lettera, per esempio **b.**

- Indicare quale dei seguenti comandi consente di modificare i permessi del file "pippo" dalla maschera rw- rw- -w- alla maschera r-- rw- r-x
  - chmod a=rx pippo ; chmod o+r-x pippo
  - chmod 660 pippo ; chmod o+r-x pippo
  - chmod 461 pippo ; chmod o+r pippo
- Siano "16 -rwxr-xr-x@ 1 marcoautili staff 4631 Jan 31 2020 apue.h" le informazioni stampate dal comando "ls -las apue.h", il numero 31 indica
  - che il file apue.h ha 31 attributi estesi ad esso associati
  - che l'id del superuser del file apue.h è 31
  - Il giorno della data di ultima modifica
- egrep '^\\+\$' < pippo.txt
  - stampa tutte le righe del file pippo.txt che contengono solo il carattere '+'
  - stampa tutte righe del file pippo.txt che contengono il carattere composto '\\+' per End-Of-Line
  - utilizza l'operatore '+' per stampare la somma dei numeri contenuti in tutte le righe
- Il comando sed -e 's/&/&/g' -e 's/ &#38;/&/g' < p.htm
  - stampa su stdout il contenuto del file p.htm sostituendo tutte le occorrenze di amp; con #38;
  - stampa su stdout il contenuto del file p.htm sostituendo tutte le occorrenze di & con & e &#38; con &
  - stampa su stdout il contenuto del file p.htm sostituendo tutte le occorrenze di & con &#38;
- Scrivere nelle due righe a destra cosa stampa il seguente script.

es5.sh	
<pre>#!/bin/bash  v1='pere' v2='marmellata'  function func () {     v3='ceci'     v4="\$1 \$v1 and \$2" }  (func pasta ceci) echo "only \$v1 and \$v3" echo "\$v2 and \$v3"</pre>	<div style="text-align: center;"> <u>only pere and</u>  <u>marmellata and</u> </div>

6. Utilizzando soltanto la riga sotto, descrivere cosa fa il seguente comando.

```
sed -e 's/^/+$/g' pippo.txt
```

Sostituisce l'inizio di ogni riga con +\$-

7. Considerare i seguenti file e la loro disposizione nella directory corrente ESERCIZIO7, all'interno della quale: la sottodirectory myobj contiene solo il file mysrc1.obj e la sottodirectory myinc contiene il file myheader.h.

ESERCIZIO7	Makefile
<div> <div>myinc</div> <div>myobj</div> <div>Makefile</div> <div>src1.c</div> <div>src2.c</div> </div> <div>mysrc1.obj</div>	<pre>myobj/my%.obj: %.c myinc/myheader.h     gcc -c -o \$@ \$&lt; -I./myinc  main1: myobj/mysrc1.obj myobj/mysrc2.obj     gcc -o \$@ \$^ -I./myinc  .PHONY: clean  clean:     rm -f myobj/my*.obj *~ core ./myinc/*~</pre>

src1.c	src2.c	./myinc/myheader.h
<pre>#include &lt;myheader.h&gt;  int main() {     myOPSLabExam();     return(0); }</pre>	<pre>#include &lt;stdio.h&gt; #include &lt;myheader.h&gt;  void myOPSLabExam(void) {     printf("It's quite easy!\n");     return; }</pre>	<pre>void myOPSLabExam(void);</pre>

Assumendo che il file src1.c non abbia subito modifiche dall'ultima compilazione andata a buon fine senza errori, scrivere nelle righe sotto (non usare tutte le righe) qual è l'output sul terminale dell'esecuzione del comando make:

```
gcc -c -o myobj/mysrc2.obj src2.c -I./myinc
gcc -o main1 myobj/mysrc1.obj myobj/mysrc2.obj -I./myinc
```

INOLTRE, scrivere nelle righe sotto (non usare tutte le righe) qual è l'output sul terminale di un'ulteriore esecuzione del comando make dopo aver modificato il file myheader.h.

```
gcc -c -o myobj/mysrc1.obj src1.c -I./myinc
gcc -c -o myobj/mysrc2.obj src2.c -I./myinc
gcc -o main1 myobj/mysrc1.obj myobj/mysrc2.obj -I./myinc
```

8. Utilizzando soltanto la riga sotto, descrivere cosa fa il seguente comando.

```
#include "apue.h"
#define PROGPATHNAME "${PROGPATHNAME:-cat -n }"

int
main(int argc, char *argv[])
{
    char    line[MAXLINE];
    FILE    *fpin, *fpout;

    if (argc != 2)
        err_quit("usage: es8 <pathname>");

    if ((fpin = fopen(argv[1], "r")) == NULL)
        err_sys("can't open %s", argv[1]);

    if ((fpout = popen(PROGPATHNAME, "w")) == NULL)
        err_sys("popen error");

    while (fgets(line, MAXLINE, fpin) != NULL) {
        if (fputs(line, fpout) == EOF)
            err_sys("fputs error to pipe");
    }

    if (ferror(fpin))
        err_sys("fgets error");
    if (pclose(fpout) == -1)
        err_sys("pclose error");

    exit(0);
}
```

Concatena i file dati in input e stampa in output il contenuto dei file con le righe numerate

9. Utilizzando soltanto le righe sotto, scrivere cosa stampa il seguente programma.

```
#include "apue.h"
#include <pthread.h>

typedef struct {
    int *ar;
    int n;
} subarray;

void *incrsun(void *arg) {
    for (int i = 0; i < ((subarray *)arg)->n; i++)
        ((subarray *)arg)->ar[i] = ((subarray *)arg)->ar[i] + i;

    return NULL;
}

int main(void) {
    int    ar[10] = {1,2,3,4,5,6,7,8,9,10}, err, i;
    pthread_t  th1, th2;
    subarray  sb1, sb2;

    sb1.ar = &ar[0];
    sb1.n = 5;
    (void) pthread_create(&th1, NULL, incrsun, &sb1);

    sb2.ar = &ar[5];
    sb2.n = 9;
    (void) pthread_create(&th2, NULL, incrsun, &sb2);
}
```

```

err = pthread_join(th1, NULL);
if (err != 0)
    err_exit(err, "can't join with thread 1");
printf("First thread joined!\n");

err = pthread_join(th2, NULL);
if (err != 0)
    err_exit(err, "can't join with thread 2");
printf("Second thread joined!\n");

for (i = 0; i < 10; i++)
    printf("%d\n", ar[i]);

exit(0);
}

```

First thread joined!
Second thread joined!
1
3
5
7
9
6
8
10
12
14

Indipendentemente dal valore di sb2.n, printf("%d\n", ar[i]); stampa sempre gli stessi valori. La cosa che cambia è la stampa degli index "i" (se presente)  
Es. se sb1.n va da 0 a 5 e sb2.n va da 5 a 13, stampa:

```

index i = 0
index i = 1
index i = 2
index i = 3
index i = 4
.
.
.
.
.
.
.
.
.
index i=12

```