

Cognome:	Nome:
Matricola:	Anno immatricolazione:
<input type="checkbox"/> Totale 6 CFU (1h) ESERCIZI dal 1 al 10	<input type="checkbox"/> Totale 3 CFU (30min) ESERCIZI dal 1 al 6
<b>PROPEDEUTICITÀ:</b> lo studente è consapevole che <b>NON PUÒ SOSTENERE</b> questo esame <b>SE NON SI È SOSTENUTO</b> l'esame di "Lab. di Programmazione di Sistema - LPS" (precedentemente, "Lab. di Architettura degli Elaboratori - LAE")	
<input type="checkbox"/> Lo studente dichiara di aver già superato l'esame di LPS (LAE).	
<input type="checkbox"/> Lo studente dichiara di voler sostenere questo esame <b>CON RISERVA SUL SUPERAMENTO DI LPS (LAE) IN UNO DEGLI APPELLI DELLA SESSIONE CORRENTE.</b> Lo studente è consapevole che <b>PERDERA' IL VOTO DI QUESTO ESAME SE NON SUPERERA' LPS (LAE) IN UNO DEGLI APPELLI DELLA SESSIONE CORRENTE.</b>	

- 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.**
- Se si sta svolgendo l'esame a distanza, per indicare la risposta scelta cambiare il **colore del testo in rosso**, oppure sottolineare, oppure **evidenziare**.

**1. Indicare quale dei seguenti comandi consente di modificare i permessi del file "pippo" dalla maschera `rwX r-x r-x` alla maschera `rw- rw- -w-`**

- `chmod a=rx pippo ; chmod o=w pippo`
- `chmod 644 pippo.rw ; chmod o=w pippo`
- `chmod 660 pippo ; chmod o=w pippo`

**2. Il comando `prog 2> pippo`**

- Redirige i messaggi di errore del comando `prog` nel file `pippo`
- Il comando `prog` inserisce il numero 2 all'inizio di ogni linea del file `pippo`
- Il comando `prog` incrementa di 2 tutti i numeri contenuti nel file `pippo`

**3. La pipeline `ls -la | grep "-.....@"`**

- stampa l'elenco di tutti i file nascosti nella directory corrente con permessi `-rwxrwxrwx@`
- stampa l'elenco di tutti i file nella directory corrente con ACL
- stampa l'elenco di tutti i file nella directory corrente con attributi estesi

**4. Il comando `grep "eur $" pippo`**

- Stampa le linee del file `pippo` che finiscono con "eur"
- Stampa le linee del file `pippo` che contengono la stringa "eur" dopo averla sostituita con "\$"
- Stampa il contenuto del file `pippo` dopo aver convertito gli "euro" in "dollari"

**5. Scrivere nella sola riga in basso a destra cosa stampa il comando `es5.sh 08`**

<p style="text-align: center;"><b>es5.sh</b></p> <pre>#!/mybin/bash  function myfunction {     s=\$1     if [ \${#s} -lt 8 ]; then         res=\$(myfunction \$1\$1)     else         res=\$1     fi     echo \$res }  res=\$(myfunction \$1) echo \$res</pre>	<hr/>
--	-------

6. Scrivere nelle tre righe sotto cosa stampa il comando `./es6.sh es6.sh es6.sh`

```
#!/bin/bash

if [ $# -lt 1 ]
then
    echo "Usage: $0 file ..."
    exit 1
fi

l=0
n=0
totl=0

for f in $*
do
    l=`wc -l < $f`
    echo "$f: $l"
    n=$((n + l))
    totl=$((totl + l))
done

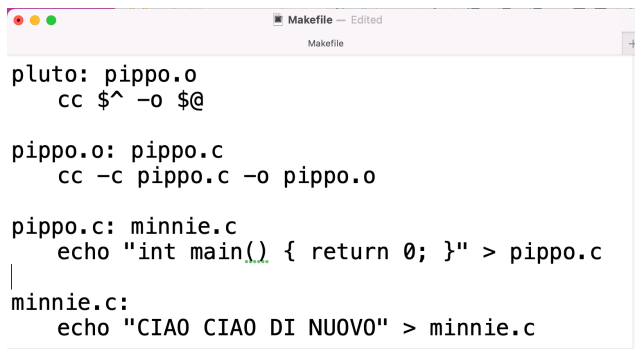
echo "n=$n in total, with totl=$totl in total"
```

---

---

---

7. Assumendo che il Makefile seguente sia l'unico file contenuto nella directory corrente, scrivere nelle righe sotto (non usare tutte le righe) qual è l'output sul terminale dell'esecuzione del comando `make`



```
pluto: pippo.o
    cc $^ -o $@

pippo.o: pippo.c
    cc -c pippo.c -o pippo.o

pippo.c: minnie.c
    echo "int main() { return 0; }" > pippo.c
|
minnie.c:
    echo "CIAO CIAO DI NUOVO" > minnie.c
```

---

---

---

---

---

**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 cancellato il file `pippo.o` precedentemente generato

---

---

---

---

---

8. Scrivere nelle righe sotto (non usare tutte le righe) cosa stampa il seguente programma passando in input es8.c su richiesta "Enter file name: "

```
es8.c
#include <stdio.h>

int main() {
    FILE *fileptr;
    int c = 0;
    char filechar[40], chr;

    printf("Enter file name: ");
    scanf("%s", filechar);
    fileptr = fopen(filechar, "r");

    chr = getc(fileptr);

    while (chr != EOF) {
        if (chr == '\n')
            c = c + 1;
        chr = getc(fileptr);
    }

    fclose(fileptr);

    printf("For file %s, c is equal to %d\n", filechar, c);

    return 0;
}
```

---

---

---

---

---

9. Completare il seguente programma C che legge i nomi delle directory entry contenute nella directory corrente e li stampa sullo standard output

```
#include ...

int main (void)
{
    DIR *dirp;
    struct _____ *direntp;
    char path[512];

    getcwd(path, 512);

    dirp = _____;

    if (_____) {
        while ((direntp = _____))
            puts(_____);

        closedir(_____);
    } else
        puts("\nErrore di apertura: ./\n");

    return 0;
}
```

10. Utilizzando le prime due righe sotto, descrivere cosa fa il seguente programma C e poi, utilizzando le rimanenti righe sotto, scrivere cosa stampa (si assuma che la prima chiamata a (int)getpid() ritorni 9087 e la seconda 9088)

```
#include ...

volatile sig_atomic_t usr_interrupt = 0;

void synch_signal (int sig) {
    usr_interrupt = 1;
}

void pronto(int signum) {
    usr_interrupt = 1;
}

void child_function (void) {
    printf ("Sono pronto!!! Il mio pid e' %d.\n", (int)getpid());
    kill (getppid(), SIGUSR1);
    printf ("Ciao, ciao\n");
    exit(0);
}

int main (void) {
    pid_t child_id;

    signal(SIGUSR1, pronto);

    printf ("Sono pronto!!! Il mio pid e' %d.\n", (int)getpid());

    child_id = fork();

    if(child_id == 0) child_function();
    while(!usr_interrupt);
    printf("Ricevuto, ciao a tutti\n");

    return 0;
}
```