

Matricola: _____

Nome: _____

Cognome: _____

ESAME Programmazione Avanzata e Parallela

12 febbraio 2024

L'esame consiste di 10 domande a risposta multipla sugli argomenti del corso. Ogni domanda può ricevere un punteggio massimo di *tre* punti. Affinché una risposta sia considerata valida la scelta *deve essere motivata*. Una risposta errata o non motivata riceverà *zero* punti.

Domanda 1

Si supponga di avere il seguente Makefile:

```
CC = gcc
CFLAGS = -O3 -Wall --std=c11 --pedantic
```

```
all: program
```

```
program: main.o bar.o foo.o
        $(CC) $^ -o $@
```

```
main.o: main.c
        $(CC) $(CFLAGS) -c $<
```

```
%.o: %.c %.h
        $(CC) $(CFLAGS) -c $<
```

```
.PHONY: clean
```

```
clean:
        rm *.o
```

Si supponga che sia stato invocato `make` e successivamente il file `foo.c` sia modificato. Alla successiva invocazione di `make` quali comandi saranno eseguiti?

```
gcc -O3 -Wall -c foo.c
☐ gcc -O3 -Wall main.o bar.o foo.o -o program
☐ gcc -O3 -Wall -c foo.c

gcc -O3 -Wall -c bar.c bar.h
gcc -O3 -Wall -c foo.c foo.h
☐ gcc -O3 -Wall -c main.c
gcc -O3 -Wall main.o bar.o foo.o -o program
```

Domanda 2

Si supponga di avere questi due frammenti di codice C:

Frammento 1

```
for (int i = 0; i < n; i++) {
    if (v[i] < 10) {
        s += v[i];
    } else {
        s += 10;
    }
}
```

Frammento 2

```
for (int i = 0; i < n; i++) {
    int q = (v[i] < 10);
    s += q * v[i] + (1 - q) * 10;
}
```

e si assuma di avere variabili s , v definite e del tipo corretto.

Quale delle seguenti affermazioni è corretta?

- | | |
|--|--|
| <input type="checkbox"/> I due codici non sono equivalenti | <input type="checkbox"/> Il frammento 2 è sempre eseguito in minor tempo del frammento 1 |
| <input type="checkbox"/> Il frammento 1 è sempre eseguito in minor tempo del frammento 2 | <input type="checkbox"/> Quale sia il frammento più veloce dipende dalla distribuzione dei valori in v |
-
-
-

Domanda 3

Sia dato il seguente frammento di codice C:

```
for (int i = 0; i < n; i++) {  
    v[i] = 0;  
    for (int j = 0; j < n; j++) {  
        v[i] += q[j * n + i];  
    }  
}
```

dove v è un vettore di n elementi e q è un vettore di n^2 elementi rappresentate una matrice quadrata di lato n in ordine *row-major*.

Quale delle seguenti affermazioni è corretta?

- ☐ Per questa tipologia di accesso una rappresentazione di q come lista concatenata con gli elementi nello stesso ordine di q avrebbe garantito maggiore località di memoria
 - ☐ L'accesso sarebbe più efficiente se q fosse in forma *column-major*
 - ☐ Il ciclo più interno può essere correttamente parallelizzato con OpenMP aggiungendo solamente `#pragma omp parallel for`
 - ☐ L'accesso sarebbe più efficiente se anche c fosse in forma *row-major*
-
-
-

Domanda 4

Dato il seguente codice che utilizza due diverse rappresentazioni per gli stessi dati (i.e., due strutture differenti):

```
#include <stdio.h>  
#include <stdlib.h>
```

```
struct S1 {  
    uint32_t a1;  
    uint64_t a2;  
    uint32_t a3;  
};
```

```
struct S2 {  
    uint32_t a1;  
    uint32_t a3;  
    uint64_t a2;  
};
```

Quale delle seguenti affermazioni è *errata*?

- | | |
|---|--|
| <input type="checkbox"/> Le due strutture possono avere dimensioni differenti | <input type="checkbox"/> Il numero di strutture in una linea di cache può essere diverso tra S1 e S2 |
| <input type="checkbox"/> Le due strutture avranno sempre lo stesso padding | <input type="checkbox"/> Le strutture potrebbero avere padding alla fine, non solo tra i diversi campi |
-
-
-

Domanda 5

Quale delle seguenti affermazioni sull'I/O è *errata*?

- | | |
|---|--|
| <input type="checkbox"/> <code>fread</code> può leggere meno del numero di elementi richiesti | <input type="checkbox"/> <code>mmap</code> inserisce EOF per indicare dove terminano i dati |
| <input type="checkbox"/> <code>fread</code> può essere più efficiente di chiamate consecutive a <code>getc</code> | <input type="checkbox"/> Se dobbiamo svolgere un accesso non sequenziale ai dati contenuti nel file possiamo utilizzare <code>fseek</code> |
-
-
-

Domanda 6

Dato il seguente codice C facente uso di OpenMP:

```
#include <stdio.h>
#include <omp.h>

float * random_matrix(int n, int m) { /* ... */ }

int main(int argc, int * argv[])
{
    const int n = 1000;
    float * M = random_matrix(n, n);
    float s = 0;
    #pragma omp parallel
    {
        float ps = 0;
        #pragma omp for collapse(2)
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                ps += M[i * n + j];
            }
        }
        #pragma omp critical
        {
            s += ps;
        }
    }
    printf("%f\n", s);
    return 0;
}
```

Si supponga che la funzione `random_matrix` sia correttamente definita e ritorni un vettore di valori floating point tra 0 e 1.

Quale delle seguenti affermazioni è corretta?

- | | |
|--|---|
| <input type="checkbox"/> Vi è una race condition dato che l'accesso a <code>M</code> non è in una sezione critica | <input type="checkbox"/> Sarebbe stato possibile aggiungere <code>nowait</code> a <code>#pragma omp for collapse(2)</code> senza modificare la correttezza del codice |
| <input type="checkbox"/> Per essere corretto <code>#pragma omp critical</code> dovrebbe essere sostituito da <code>#pragma omp single</code> | <input type="checkbox"/> La variabile <code>ps</code> dovrebbe essere definita prima di <code>#pragma omp parallel</code> |
-
-
-

Domanda 7

Dato il seguente codice Python:

```
class A:
```

```
    def g(self, x):  
        self.f(x + 1)
```

```
    def f(self, x):  
        print(f"A {x}")
```

```
class B(A):
```

```
    def g(self, x):  
        self.f(x + 2)
```

```
class C(B):
```

```
    def f(self, x):  
        print(f"C {x}")
```

```
class D(C):
```

```
    def g(self, x):  
        super().g(x + 1)
```

```
x = D()
```

```
x.g(1)
```

quale è il valore risultante dall'ultima riga (i.e., `x.g(1)`)?

☐ A 2

☐ C 3

☐ C 4

☐ Viene generata una eccezione perché tutte le classi devono implementare sia `g` che `f`

Domanda 8

Dato il seguente codice Python:

```
def f(g, x):  
    def h1(y):  
        return x + 2*y  
  
    def h2(z):  
        return g(h1(x))  
  
    return h2
```

func = f(**lambda** x: x + 2, 6)

Quale è il valore ritornato da func(3):

- ☐ Non è possibile usare lambda come argomenti di funzioni ☐ 20
- ☐ 18 ☐ 8

Domanda 9

Dato il seguente codice Python:

```
class StrangeException (Exception):  
    pass
```

```
class UnusualException (Exception):  
    pass
```

```
def f(x):  
    try:  
        g(x)  
    except Exception:  
        print("A")  
    except StrangeException:  
        print("B")
```

```
def g(x):  
    try:  
        if (x < 0):  
            raise StrangeException()  
        if (x > 10):  
            raise UnusualException()  
    except UnusualException:  
        print("C")
```

f(-3)

f(30)

Cosa viene stampato a schermo al termine dell'esecuzione?

☐ A C

☐ C C

☐ B C

☐ Viene stampato un messaggio di errore perché l'eccezione `StrangeException` non è catturata

Domanda 10

Dato il seguente codice Python:

```
def f(x):  
    while True:  
        yield lambda y: x + y  
        x += 1
```

```
h = f(0)  
for i in range(5):  
    g = next(h)  
    print(g(i))
```

Quali sono i valori stampati a schermo?

☐ <function f.<locals>.<lambda> at 0x104eef7f0>
☐ <function f.<locals>.<lambda> at 0x104eef760>
...

☐ 0 1 2 3 4

☐ Viene generata una eccezione dato che non è possibile chiamare g

☐ 0 2 4 6 8
