1. Record di attivazione A

Scrivi una funzione in **C** che dato in ingresso un numero n calcola l'n-esimo numero di Fibonacci.

La **successione di Fibonacci** è una successione di numeri interi naturali definibile assegnando i valori dei due primi termini, $F_0 := 0$ ed $F_1 := 1$, e chiedendo che per ogni successivo sia $F_n := F_{n-1} + F_{n-2}$ con n>1.

Scrivi tre versioni: una non ricorsiva, una ricorsiva senza tail recursion e una ricorsiva con tail recursion. Passa i parametri per valore e restituisci il valore del risultato. Non usare alcuna variabile globale. Scrivi anche un main di esempio in cui chiami la funzione per n = 3 e assegni il risultato ad una variabile locale al main (FIB_3).

Disegna il record di attivazione per tutte e tre le versioni fino alla massima estensione del record di attivazione. Nel caso di tail recursion, spiega quali ottimizzazioni hai adottato o potresti adottare.

2. Record Attivazione B

Considera questo codice C:

```
void print_facts(int num1, int num2);
int max_of_two(int j, int k);
double avg_of_two(int c, int d);

int main(void)
{
  int i;
  int j;
  i = -8;
  j = 7;
  print_facts(i, j);
  return 0;
}

void print_facts(int num1, int num2)
{
  int larger;
  double the_avg;
  larger = max_of_two(num1, num2);
  the_avg = avg_of_two(num1, num2);
  printf("given two integers %d and
%d,\n", num1, num2);
  printf("the larger is %d and the average
is %g.\n",
         larger, the_avg);
}

int max_of_two(int j, int k)
{
  if (j < k)
    j = k;
  return j;
}

double avg_of_two(int c, int d)
{
  double sum;
  sum = c + d;
  return (c + d) / 2.0;
}
```

Scrivi il record di attivazione fino alla sua massima estensione nei due metodi max_of_two e avg_of_two

3. Cyclone

A cosa servono i puntatori notnull in Cyclone? Fa un esempio in cui spieghi.

4. C++

Considera il seguente codice?

```
class Veicolo {
 private:
   int age() { return 1; }
 public:
   int power() { return 2; }
   virtual int color() { return 3; }
};

class Auto: private Veicolo {
 public:
   int color() { return 5; }
};
```

```
class Moto: public Veicolo {
public:
  int power() { return 6; }
  int color() { return 7; }
};
int main() {
 1.    Veicolo p;
 2.    cout << p.age() << endl;
 3.    cout << p.power()<< endl;

 4.    Auto s;
 5.    cout << s.age() << endl;
 6.    cout << s.power()<< endl;
 7.    cout << s.color()<< endl;

 8.    Moto pr;
 9.    cout << pr.age() << endl;
 10.   cout << pr.power()<< endl;
 11.   cout << pr.color()<< endl;

 12.   Auto * ps = &p;
 13.   cout << ps->color()<< endl;
 14.   Veicolo * pp = &s;
 15.   cout << pp->color()<< endl;

 16.   Veicolo * pp2 = &pr;
 17.   cout << pp2->color()<< endl;

 18.   Moto * ppr = &pr;
 19.   cout << ppr->color()<< endl;

 20.   Moto * ppr2 = &p;
 21.   cout << ppr2->color()<< endl;

 22.   Veicolo pr3 = pr;
 23.   cout << pr3.age() << endl;
 24.   cout << pr3.power()<< endl;
 25.   cout << pr3.color()<< endl;

 26.   Auto s3 = p;
 27.   cout << s3.age() << endl;
 28.   cout << s3.power()<< endl;
 29.   cout << s3.color()<< endl;
     }
```

Quale è l'output/effetto prodotto da ogni linea numerata del main? Se contiene un errore scrivi errore, spiega l'errore e ignora la linea (ed eventuali istruzioni che dipendono da essa).

5. Template in C++

A cosa servono i template in C++? Fai un esempio di classe generica e di metodo generico.

6. Dinamic Binding in Java

Date le seguenti dichiarazioni:

```
class A{
   public int m(long d){return 1;}
   public int f(double d){return 2;}
}

A va = new B();
```

```
class B  extends A{
   public int m(short d){return 3;}
   public int f(double d){return 4;}
   public int f(float f){return 5;}
}
```

Quale è il valore delle seguenti tre espressioni spiegando bene (cioè anche il processo di early e late binding dove necessario) il perché (anche se le ritieni errate):

```
1. va.m(2);

2. va.m((short)2);

3. va.m(2.0);

4. ps.f(2.5f);

5. ps.f(2.5);
```

7. Java – visitor pattern

A cosa seve il visitor pattern? Illustra mediante un esempio.

8. Semantica assiomatica

Scrivi un piccolo programma che calcola x * y , utilizzando solo l'operazione di decremento (--) e incremento (++). Scrivi le precodizioni e le postcondizioni e cerca di dimostrare la correttezza.

SOLUZIONI

fibonacci con tail

```
fib(i, current = 0, next = 1):
  if i == 0:
    return current
  else:
    return fib(i - 1, next, current + next)

fib(i)
return fib(i,0,1)
```