Informatica III – 9 /2/ 12 – info 3 – Nome

Record di attivazione A

Scrivi una funzione **`isordered`** in **C** che dato in ingresso un array di interi restituisce true (1) se l'array è ordinato in ordine crescente altrimenti restituisce false. Al solito, scrivi tre versioni: una non ricorsiva, una ricorsiva senza tail recursion e una ricorsiva con tail recursion.

Specifica esattamente i parametri che passi alla procedura, il tipo di passaggio utilizzato e il loro significato.

Scrivi anche un main di esempio in cui chiami la funzione con un array (corto) di esempio e invocala in modo assegni il risultato ad una variabile globale. Non usare alcuna altra variabile globale.

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.

Recordi di attivazione in C++

Considera questo codice C++:

```
int A = 15;

int foo(int &y){
     y++;
     if (y > 0 ){
       int q = 10;
       A = y;
       return g(y);
     }else{
       int y = 90;
       return foo(y);
     }
}
int g(int A){
    int x = 10;
    return A + x;
}

int main(){
      int q = -50 + A;
      cout << "1. " << m(q)<< endl;
      cout << "2. " << q<< endl;
      cout << "3. " << A<< endl;
}
```

Quale è l'output? Scrivi il record di attivazione per la chiamata delle due funzioni.

Cyclone

A cosa servono i puntatori fat? Come si dichiarano? Come si usano? Fai un piccolo esempio. Quali sono i vantaggi e svantaggi del loro uso?

Tipi opachi in C

Come si usano i tipi opachi in C? A cosa servono? Fai un piccolo esempio.

C++

Considera il seguente codice?

```
class Computer {
 private:
   int pri() { return 1; }
 public:
   int pub() { return 2; }
   virtual int vpub() { return 3; }
};

class Smartphone: private Computer {
 public:
   int vpub() { return 5; }
};

class Laptop: public Computer {
 private:
  int pri() { return 6; }
 public:
  int vpub() { return 7; }
};

int main() {
 1.    Computer p;
 2.    cout << p.pri() << endl;
 3.    cout << p.pub()<< endl;
 4.    cout << p.vpub()<< endl;

 5.    Smartphone s;
 6.    cout << s.pri() << endl;
 7.    cout << s.pub()<< endl;
 8.    cout << s.vpub()<< endl;

 9.    Laptop m;
 10.   cout << m.pri() << endl;
 11.   cout << m.pub()<< endl;
 12.   cout << m.vpub()<< endl;

 13.   Smartphone * pv = &p;
 14.   Laptop * pm = &p;

 15.   Computer * pp = &s;
 16.   cout << pp->pri()<< endl;
 17.   cout << pp->pub()<< endl;
 18.   cout << pp->vpub()<< endl;

 19.   Computer * pp2 = &m;
 20.   cout << pp2->pri()<< endl;
 21.   cout << pp2->pub()<< endl;
 22.   cout << pp2->vpub()<< endl;

 23.   Computer e1 = s;
 24.   cout << e1.pri() << endl;
 25.   cout << e1.pub()<< endl;
 26.   cout << e1.vpub()<< endl;

 27.   Computer e2 = m;
 28.   cout << e2.pri() << endl;
 29.   cout << e2.pub()<< endl;
 30.   cout << e2.vpub()<< 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).

Dinamic Binding in Java

Date le seguenti dichiarazioni:

```
class Computer{
   public int m(Computer d){return 5;}
}

Object oc = new Computer();
Computer cs = new Smartphone();
```

```
class Smartphone  extends Computer{
   public int m(Computer d){return 6;}
   public int m(Smartphone d){return 7;}
}
Smartphone ss = new Smartphone();
```

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):

```
oc.equals(oc);
oc.equals(cs);
cs.m(oc);
cs.m(cs);
```

```
cs.m(ss);
ss.m(oc);
ss.m(cs);
ss.m(ss);
```

Java

Spiega l'uso delle wildcard per generici.

Testing – 6 crediti

Testing programmi

Dato il seguente programma:

```
public boolean isLeapYear(int year) {
 if (((year % 4 == 0) && (year % 100 != 0) && (year >= 1) || (year % 400 == 0)))
  return true;
 else
  return false;
}
```

che calcola se un anno è bisestile, disegna il grafo del programma e trova i casi di test per la copertura delle istruzioni, delle decisioni, delle condizioni e l'MCDC

Testing FSM

Scrivi una macchina a stati finiti con almeno 3 stati e trovane un transition tuor (anche non euleriano). Introduci un difetto di output e scopri se il tuo test è in grado di scoprirlo.

KeyHoare – 2/ 12 – info 3 – Nome _____

Istruzioni:

- scarica lo zip dalla directory indicata dal professore e unzippala sul desktop
- per farlo partire, doppio click su startKeYHoare.bat
- carica il tuo file (se hai errore chiudi, correggi e riapri)
- salva la prova con File -> Save .proof.

Problema A:

Scrivi un programma che calcola x+y con soli incrementi e decrementi di una unità. Dovrebbe funzionare sia che x e y siano maggiori o minori di zero. Metti il risultato nella variabile z. Dovrebbe essere del tipo

```
z = x;
while ( *** ) {
  if (y > 0){
     *** (solo + o -1)
  } else{
     *** (solo + o -1)
  }
}
```

Scrive le pre e post condizioni e dimostra la correttezza. Aiuto, se devi dire a diverso da b in keyhoare si scrive (! (a==b)).