Nom i cognoms:

Algorísmia i Programació 1, GCED, 25 de gener de 2018

L'examen dura dues hores i mitja. Es valorarà la concisió, claredat i brevetat a més de la completesa i l'exactitud de les respostes. Contesteu a l'espai marcat amb bona lletra. Poseu el vostre nom a cada full. No podeu consultar cap material addicional.

1 Pupurri (2 punts)

1. Digueu quin és l'algorisme que més us ha agradat aprendre en aquest curs i perquè.

2. Expliqueu què és un compilador.

3. Expliqueu què calcula el garbell d'Eratòstenes.

4. Expliqueu la diferència entre els paràmetres reals i els paràmetres formals.

5. Expliqueu com triar si els paràmetres s'han de passar per valor, per referència o per referència constant (en C++).

6. Considereu un tipus enter sense signe que ocupa 8 bits.

   - Quants valors diferents pot representar?
   - Quin és el valor més petit que pot representar?
   - Quin és el valor més gran que pot representar?

7. Què escriuen aquests programes en C++ i Python3?

```
void foo (int& a, int& b) {
    a -= b / 2;
    b += a;
    cout << a << " " << b << " ";
}

int main () {
    int a = 3, b = 5;
    foo(b, a);
    cout << a << " " << b << endl;
}
```

```
def foo (a, b):
    a -= b / 2
    b += a
    print(a, b, end="")

def main ():
  a, b = 3, 5
  foo(b, a)
  print(a, b)

main()
```

8. Amb quina comanda compilareu el vostre programa `p.cc` en C++ aquesta tarda?

Nom i cognoms:

2 Operacions geomètriques (2 punts)

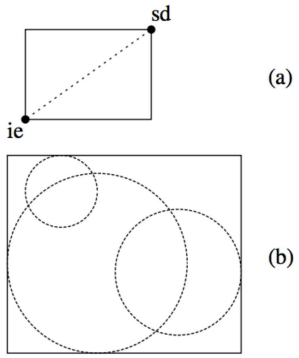
Considereu les definicions de tipus següents:

```
struct Punt {
    double x, y;
};

struct Rectangle {
    Punt ie, sd;
};

struct Cercle {
    Punt c;
    double r;
};

using Cercles = vector⟨Cercle⟩;
```



Un punt a $\mathbb{R}^2$ té dues coordenades (x, y). El tipus *Rectangle* enmagatzema un rectangle rectilini (és a dir, que té les arestes paral·leles als eixos) i es representa amb dos punts: els extrems inferior-esquerre (*ie*) i superior-dret (*sd*) de la diagonal, tal com mostra la figura (a). Un cercle es representa amb el seu centre (*c*) i el seu radi (*r*).

Definiu i implementeu funcions i/o accions per a cadascuna de les tasques següents:

1. Dir si un punt es troba dins d'un cercle.
2. Retornar el rectangle rectilini més petit que conté un cercle donat.
3. Aplicar una translació δx,δy a un rectangle.
4. Retornar un punt aleatori dins d'un rectangle, tot seguint una distribució uniforme.
5. Retornar un punt aleatori dins d'un cercle, tot seguint una distribució uniforme.
6. Retornar el rectangle rectilini més petit que conté tots els cercles d'un vector, tal com mostra l'exemple de la figura (b).

Podeu utilitzar funcions matemàtiques de la llibreria estàndard com ara *sin()*, *sqrt()*,... i la funció *rand()*, que retorna un enter aleatori uniformement distribuït en 0..*RAND_MAX*.

Nom i cognoms:

3 Ordenació per bombolla (2 punts)

La funció següent implementa un algorisme d'ordenació anomenat *ordenació per bombolla* per ordenar un vector d'n reals:

```
void bubble_sort (vector<double>& v) {
    int n = v.size();
    for (int i = 0; i < n − 1; ++i) {
        for (int j = n − 1; j > i; −−j) {
            if (v[j − 1] > v[j]) {
                swap(v[j − 1], v[j]);
            }
        }
    }
}
```

1. Expliciteu (potser de forma gràfica) quin és l'invariant del bucle de les i.

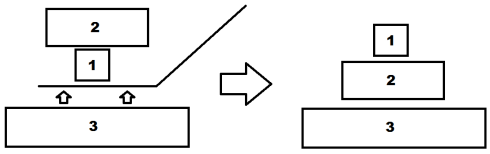
2. Compteu exactament quantes comparacions entre reals realitza aquests algorisme en funció d'n.

3. Quantifiqueu el temps d'execució d'aquest algorisme utilitzant notació asimptòtica.

4 Ordenació de creps (*crêpes, pancakes*) (2 punts)

Entrada

Sortida



Gir

Tenim n creps, de mides 1 a n (totes diferents) que estan apilades les unes damunt de les altres en qualsevol ordre damunt d'un plat. L'única operació que podem fer amb elles és inserir una espàtula entre dues creps (o entre la darrera crep i el plat) i girar totes les creps que hi hagi al damunt de l'espàtula (*hop!*). Es vol ordenar totes les creps, amb la més gran a baix.

Per exemple, la seqüència següent mostra com ordenar 6 creps de mides 1, 2, 3, 6, 4, i 5 de dalt a baix amb 4 girs. El separador | mostra el punt on es col·loca l'espàtula per aplicar el següent gir.

$$1236|45 \longrightarrow 632145| \longrightarrow 54123|6 \longrightarrow 321|456 \longrightarrow 123456.$$

Expliqueu (sense donar codi) un algorisme per ordenar n creps, expliciteu el seu invariant i digueu quants girs d'espàtula requereix en el cas pitjor (no cal que busqueu de minimitzar el nombre de girs).

Nom i cognoms:

5 Depuració de codi (2 punts)

Una biblioteca utilitza el tipus següent per emmagatzemar la seva col·lecció de llibres:

```
struct Llibre {
    string titol;
    string autor;
    int pagines;
};
```

El programa de la pàgina següent hauria de llegir n llibres i escriure'ls ordenadament. Concretament, es vol que els llibres apareguin ordenats per autor (alfabèticament). En cas d'empat, han d'aparèixer ordenats per títol (també alfabèticament) i, en cas de nou empat, per nombre de pàgines (en ordre decreixent). Fixeu-vos que un mateix llibre pot estar repetit i/o tenir diferents edicions, amb diferents nombre de pàgines.

1. El codi conté alguns errors de programació. Identifiqueu-los (amb una **E**) i esmeneu-los.
2. El codi també conté algunes aspectes menors que, malgrat no ser errors, admeten millores. Identifiqueu-los (amb una **M**), i esmeneu-los.

- Anoteu directament el codi a la pàgina següent. Unes petites edicions locals són suficients, seguiu l'exemple donat com a referència.

```
vector⟨Llibre⟩ llegir () {
    Llibre llibre;
    int n;
    cin ≫ n;
    vector⟨Llibre⟩ llibres(n);
    for (int i = 0; i ≤ n; ++i) {
        cin ≫ llibre.titol ≫ llibre.autor ≫ llibre.pagines;
        llibres.push_back(llibre);
    }
    return llibres;
}
```

```
void comp (Llibre l1, Llibre l2) {
    if (l1.titol ≠ l2.titol) return l1.titol ≤ l2.titol;
    if (l1.autor ≠ l2.autor) return l1.autor ≤ l2.autor;
    return (l1.pagines ≥ l2.pagines);
}
```

⟵ M: no calen parentesis

```
void ordena (vector⟨Llibre⟩ llibres) {
    sort(llibres.begin(), llibres.end(), comp);
}
```

```
void escriure (vector⟨Llibre⟩ llibres) {
    for (Llibre llibre : llibres) {
        cout ≪ llibre.titol ≪ " " ≪ llibre.autor ≪ " " ≪ llibre.pagines ≪ endl;
    }
}
```

```
void main() {
    vector⟨Llibre⟩ llibres = llegir();
    ordena(llibres);
    escriure(llibres);
}
```