KANDIDATNUMMER:

EKSAMEN

EMNENAVN: Matematikk for spillprogrammering

EMNENUMMER: REA2061

EKSAMENSDATO: 11.06.15

KLASSE: Bachelor Spillprogrammering

TID: 9-14

EMNEANSVARLIG: Bernt Tore Jensen (tlf. 46250024)

ANTALL SIDER UTLEVERT: 5 sider (inkludert denne)

TILLATTE HJELPEMIDLER: Godkjent kalkulator, alle skrevne og trykte hjelpemidler

INNFØRING MED PENN.

Ved innlevering skilles hvit og gul besvarelse og legges i hvert sitt omslag.

Oppgavetekst, kladd og blå kopi beholder kandidaten.

Husk kandidatnummer på alle ark.

There are 5 problems, each contributing 20 percent towards your grade. Explanations and calculations must be included.

Problem 1:

a) Two dice are thrown, one with 4 sides numbered 1 to 4, and the other with 6 sides numbered 1 to 6.

i) Find the probability of getting two 4's.

ii) Find he probability that neither dice shows 3.

b) Six dice are thrown. All six dice have 6 sides numbered 1 to 6.

i) Find the probability that the sum is 8.

ii) Find the probability that there are at least two 5's and at least two 6's.

Problem 2:

a) Show how to find the 4×4-matrix which rotates $30°$ around the line through the origin with direction $(6, 3, 1)^T$. You need to set up the appropriate matrix product in the correct order, but you do not need to do the final matrix calculation. Include figures and explanations.

b) Use quaternions to rotate the point $(0, 0, 1)$ $90°$ around the x-axis. Include all calculations.

c) The quaternion

$$q = 0.8660 + 0.3535i + 0.3535j$$

is used to perform rotation in 3-dimensional space. Find the angle of rotation and the direction of the line of rotation.

Problem 3:

Let $P_0 = (0,0), P_1 = (5,0), P_2 = (0,5), P_3 = (5,5)$ be four points in the plane. The function

```
vector2f calculatePoint(float t);
```

takes as input a number t between 0 and 1, and returns the corresponding point on the cubic Bezier curve given by $P_0, \cdots, P_3$.

a) Implement the function.

b) Calculate the point on the Bezier curve at $t = 0.2$.

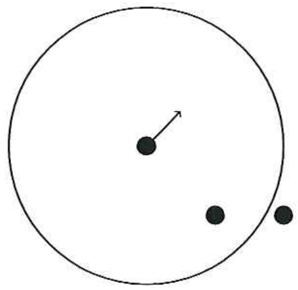
Problem 4.

Boids move around on a flat surface. A boid's position and velocity are given by two variables

```
vector2f pos;
vector2f vel;
```

a)

A boid is visible to another boid if the distance between them is less than 100. So in the picture below, only the boid within the circle of radius 100 is visible to the boid in the center of the circle.



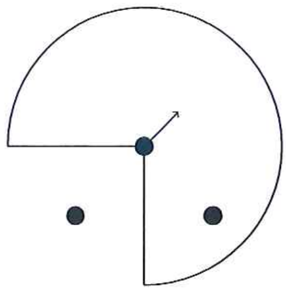
The class describing a boid contains the method

```
bool boid::isVisible(boid otherBoid);
```

which returns true if otherBoid is visible, and false otherwise. Implement the method.

b)

A boid now has a restricted field of view, with a blind spot at $\pm 45°$ in the opposite direction of the direction of flight. So in the picture below, both boids have distance less than 100 from the boid in the center, but only one boid is visible.



Implement the method

```
bool boid::isVisible(boid otherBoid);
```

taking the field of view into consideration.

c)

A collection of boids is stored as

```
boid myBoids[100];
```

Implement the rule which aligns a boid's velocity to the velocity of all visible boids.

Problem 5:

a)

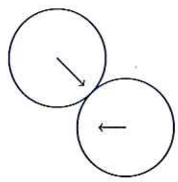
A circular object has properties

```
vector2f pos;
vector2f vel;
float invMass;  // 1/mass
// and other properties not needed for this task
```

Implement the change in velocites which occur if two such objects collide. You do not need to check for collision and you can assume the collision is elastic.

b)

A collision has occured as in the picture



where the two objects have properties

```
pos = vector2f(0,0);
vel = vector2f(0.4,-0.4);
invMass = 1;
```

and

```
pos = vector2f(1,-1);
vel = vector2f(-0.4,0);
invMass = 1;
```

Calculate the velocities after collision. You can assume the collision is elastic.