

./a-aire-disque.py

```
import math

def principale():

    # initialisation de l'état du système
    rayon = 1

    while rayon < 5:

        # affichage de l'état du système
        aire_disque = math.pi * rayon**2
        print("L'aire avec rayon", rayon, "est", aire_disque)

        # mise à jour de l'état du système
        rayon = rayon + 0.03

principale()
```

./b-diag.py

```
import math

def principale():

    # initialisation de l'état du système
    largeur = 1
    longueur = 1

    while largeur < 4:

        # affichage de l'état du système
        diagonale = (largeur**2 + longueur**2)**0.5
        print("La diagonale vaut", diagonale)

        # mise à jour de l'état du système
        largeur = largeur + 0.03
        longueur = longueur + 0.09

principale()
```

./c-disque-graphique.py

```
import math
from qtido import *

def principale():

    f = creer(800, 600)

    # initialisation de l'état du système
    rayon = 1

    while not est_fermee(f):

        # affichage de l'état du système
        effacer(f)
        couleur(f, 0, 1, 0)
        disque(f, 400, 300, rayon)

        # attente des événements etc
        attendre_pendant(f, 10) # 10 millisecondes

        # mise à jour de l'état du système
        rayon = rayon + 0.03

principale()
```

./d-disque-unite.py

```
import math
from qtido import *

def principale():

    f = creer(800, 600)
    pixel_par_cm = 38 # px / cm

    # initialisation de l'état du système
    rayon = 1 # cm
```

```

while not est_fermee(f):

    # affichage de l'état du système
    effacer(f)
    couleur(f, 0, 0, 1)
    disque(f, 400, 300, rayon*pixel_par_cm)

    # attente des événements etc
    attendre_pendant(f, 10) # 10 millisecondes

    # mise à jour de l'état du système
    rayon = rayon + 0.03

principale()

```

./e-mru.py

```

import math
from qtido import *

def principale():

    f = creer(800, 600)
    px_par_m = 10 # px / m

    # initialisation de l'état du système
    x = 0 # m

    v = 5 # m/s
    dt = 0.010 # s
    r = 2 # m

    while not est_fermee(f):

        # affichage de l'état du système
        effacer(f)
        couleur(f, 0, 0, 1)
        disque(f, 400 + x*px_par_m, 300, r*px_par_m)

        # attente des événements etc
        attendre_pendant(f, dt*1000) #ms

        # mise à jour de l'état du système
        x = x + v * dt

principale()

```

./f-mrua.py

```

import math
from qtido import *

def principale():

    f = creer(800, 600)
    px_par_m = 10 # px / m

    # initialisation de l'état du système
    y = 0 # m
    v = 20 # m/s

    dt = 0.010 # s
    r = 2 # m
    g = 9.81 # m/s²

    while not est_fermee(f):

        # affichage de l'état du système
        effacer(f)
        couleur(f, 0, 0, 1)
        disque(f, 400, 300 - y*px_par_m, r*px_par_m)

        # attente des événements etc
        attendre_pendant(f, dt*1000) #ms

        # mise à jour de l'état du système
        y = y + v * dt
        v = v + (-g) * dt

principale()

```