```
# encoding: utf-8

'''
Simulador de cibercafé
Modo de uso: TP6.py <G> <W>
Donde G y W son variables de control:
 G: cantidad de máquinas para juegos (gaming PCs)
 W: cantidad de estaciones de trabajo (workstations)
'''

import sys, random
HV = float("inf")

def VariablesDeControl():
	global G, W

	try:
		if len(sys.argv) == 3:
			G = int(sys.argv[1])
			W = int(sys.argv[2])
		else:
			G = int(raw_input('G='))
			W = int(raw_input('W='))
	except ValueError:
		print 'Error: G y W deben ser valores enteros.'
		exit(1)

def CondicionesIniciales():
	global T, TF, TPLL, TPSW, TPSG, NSW, NSG, CLL, \
			ITOW, ITOG, STOW, STOG, CARRW, CARRG, WAG
```

```
    T = 0
    TF = 13140000

    TPLL = 0
    TPSW = [HV for _ in xrange(W)]
    TPSG = [HV for _ in xrange(G)]

    NSW = NSG = CLL = 0

    ITOW = [0 for _ in xrange(W)]
    ITOG = [0 for _ in xrange(G)]

    STOW = ITOW[:]
    STOG = ITOG[:]

    CARRW = CARRG = 0
    WAG = 0

def MinTPSWorker():
    return TPSW.index(min(TPSW))

def MinTPSGamer():
    return TPSG.index(min(TPSG))

def HVTPS(TPS):
    return TPS.index(HV)

def GenerarIA():
R = random.random()
    TA = int(5 + R * 20)
    return TA
```

```
def GenerarTAG():
    R = random.random()
    TA = int(60 + R * 120)
    return TA

def GenerarTAW():
    R = random.random()
    TA = int(20 + R * 70)
    return TA

def ArrepentimientoWorker():
    global CARRW
    ARR = NSW - W > 3
    if ARR: CARRW += 1
    return ARR

def ArrepentimientoGamer():
    global CARRG
    ARR = NSG - G > 5
    if ARR: CARRG += 1
    return ARR

def EntraWorker():
    global NSW, CLL, TPSW, STOW
    NSW += 1
    CLL += 1
    if NSW <= W:
        i = HVTPS(TPSW)
        TA = GenerarTAW()
        TPSW[i] = T + TA
        STOW[i] += T - ITOW[i]
```

```
def Worker():
    global WAG
    if NSW >= W:
        if NSG < G:
            R = random.random()
            if R < 0.20:
                WAG += 1
                Gamer()
            else:
                ARR = ArrepentimientoWorker()
                if not ARR:
                    EntraWorker()
        else:
            ARR = ArrepentimientoWorker()
            if not ARR:
                EntraWorker()
    else:
        EntraWorker()


def Gamer():
    global NSG, CLL, TPSG, STOG
    ARR = ArrepentimientoGamer()
    if not ARR:
        NSG += 1
        CLL += 1
        if NSG <= G:
            i = HVTPS(TPSG)
            TA = GenerarTAG()
            TPSG[i] = T + TA
            STOG[i] += T - ITOG[i]
```

```
def LlegaCliente():
    global T, TPLL
    T = TPLL
    IA = GenerarIA()
    TPLL = T + IA

    R = random.random()
    Gamer() if R < 0.3 else Worker()

def SaleW(i):
    global T, TPSW, NSW, ITOW
    T = TPSW[i]
    NSW -= 1

    if NSW < W:
        ITOW[i] = T
        TPSW[i] = HV
    else:
        TA = GenerarTAW()
        TPSW[i] = T + TA

def SaleG(i):
    global T, TPSG, NSG, ITOG
    T = TPSG[i]
    NSG -= 1

    if NSG < G:
        ITOG[i] = T
        TPSG[i] = HV
    else:
        TA = GenerarTAG()
        TPSG[i] = T + TA

def ImprimirResultados():

    SSTOW, SSTOG = sum(STOW), sum(STOG)
    PTOW = 100 if SSTOW==0 else (1.0*sum(STOW)/len(STOW)) * 100.0/T
    PTOG = 100 if SSTOG==0 else (1.0*sum(STOG)/len(STOG)) * 100.0/T

    PPAW = CARRW * 100.0 / (CLL + CARRW + CARRG)
    PPAG = CARRG * 100.0 / (CLL + CARRW + CARRG)

    print "Resultados de la simulacion:"
    print "- Tiempo ocioso:"
    print "  PTOW = %5.1f%%" % PTOW
    print "  PTOG = %5.1f%%" % PTOG
    print "- Arrepentidos:"
    print "  PPAW = %5.1f%%" % PPAW
    print "  PPAG = %5.1f%%" % PPAG
    print "- Workers que se pasaron a Gamers:"
    print "  WAG = %d" % WAG

if __name__ == "__main__":
    VariablesDeControl()
    CondicionesIniciales()

    print "Simulando con G=%d, W=%d..." % (G, W)

    while True:
        i = MinTPSWorker()
        j = MinTPSGamer()

        if TPSW[i] < TPSG[j]:
            if TPLL <= TPSW[i]:
                LlegaCliente()
            else:
                SaleW(i)
        else:
            if TPLL <= TPSG[j]:
                LlegaCliente()
            else:
                SaleG(j)

        if T <= TF: continue
        if NSW + NSG > 0:
            TPLL = HV
            continue
        break

    ImprimirResultados()
```