In [4]:
```
using PyPlot, Random, LinearAlgebra, Optim   # Packages needed

```

In [5]:
```
charstr = """
          000000  000000  000000  00..00
          000000  000000  000000  00..00
          0.00.0  00..00  ..00..  00..00
          0.00.0  00..00  ..00..  000000
          0....0  000000  ..00..  00..00
          0....0  00..00  ..00..  00..00
          0... 0  00..00  ..00..  00..00
          """

training = reshape(collect(charstr), :, 7)
training = Int.(training[[1:6;9:14;17:22;25:30],:] .== '0')
training = reshape(training', 7*6, 4)
target = [0 0; 0 1; 1 0; 1 1]'
mapstr = "MATH";
```

In [6]:
```
function plot_chars(images)
    gray()
    n_images = size(images,2)
    for j = 1:n_images
        subplot(ceil(Int, n_images/4), 4, j)
        im = 1 .- reshape(images[:,j], 7, 6)
        imshow(im); axis("off");
    end
end
plot_chars(training)
```

Problem 1

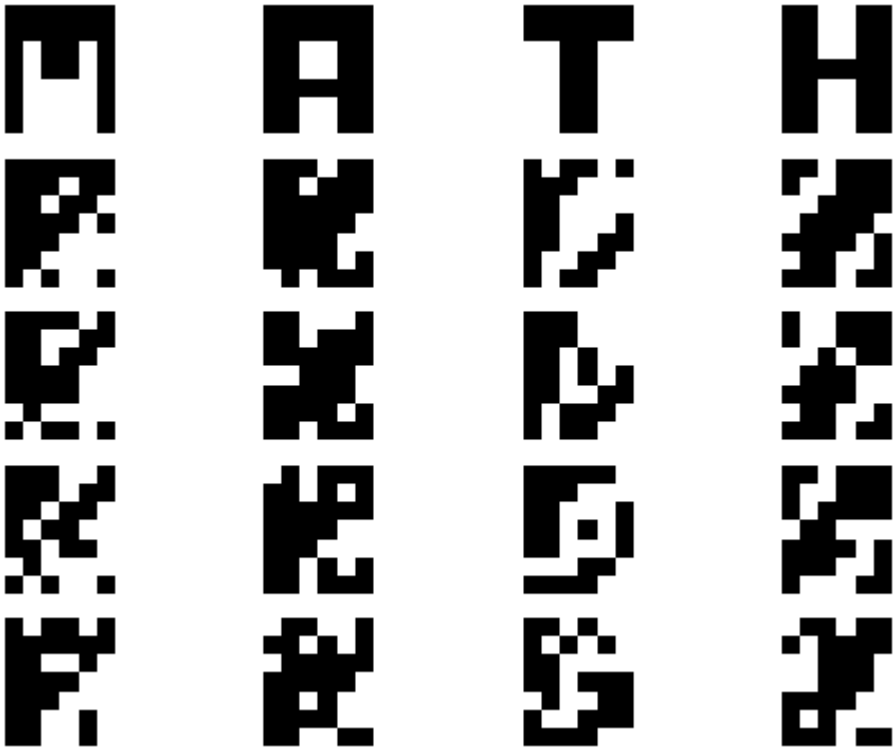
In [7]:

```
function make_testdata(training)
 testdata = zeros(42, 20)

 # Add original training images
 testdata[:, 1:4] = training

 # Generate noisy images with increasing perturbation
 for i in 1:4
 n_flips = 2 * i

 for j in 1:4
 image = reshape(training[:, j], 7, 6)
 pixels_to_flip = randperm(42)[1:n_flips]
 image[pixels_to_flip] = 1 .- image[pixels_to_flip]
 testdata[:, i*4+j] = reshape(image', 42)
 end
 end

 return testdata
end

```

Out[7]: make_testdata (generic function with 1 method)

In [8]:

```
testdata = make_testdata(training);
plot_chars(testdata)
```



Problem 2

In [11]:
```
function σ(x)
 return 1/(1+exp(0.5-x))
end
function train_sgd(; maxiter=10000, rate=1)
 V=randn(10,42)
 W=randn(2,10)
 for iteration = 1:maxiter
 j=rand(1:4)
 k=rand(1:2)
 xj= training[:,j]
 Wk=W[k:k,:]
 r = σ.(V*xj)
 y = σ.(W*r)[k]
 q = (y-target[k,j])*y*(1-y)
 u = Wk'.*r.*(1 .-r)
 dwk = q*r'
 dv = q*u*xj'
 V .-= dv.*rate
 W[k:k,:] .-= dwk.*rate
    end
    return V,W
end
```

Out[11]:
```
train_sgd (generic function with 1 method)
```

In [12]:
```
V,W = train_sgd()
```

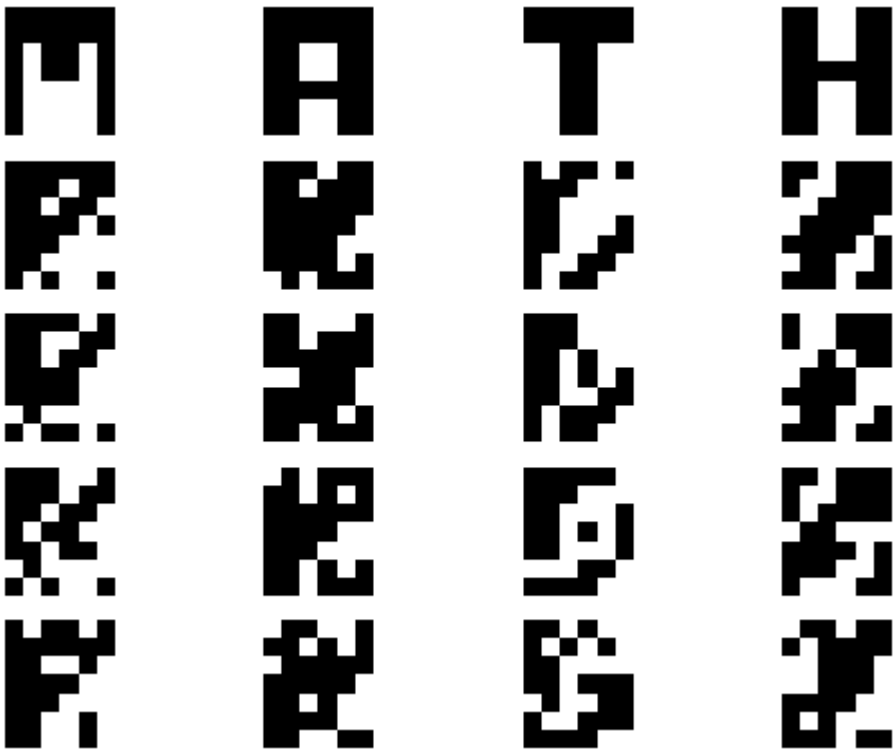
Out[12]:
```
([0.3660164740496111 -1.9837096432369477 … -0.1038500218803723 1.04740329
39624054; 0.3711166212475576 -0.5681098066131308 … 0.03495773179804071 1.
1213286101338045; … ; 0.48339136464685295 0.2448375324825795 … -0.8404280
645638401 1.3138382519772973; 0.04435789272833523 -1.7870840548025912 … -
0.1259019132364534 -0.17285192314201672], [0.20482872153935122 -4.4725268
59083275 … -2.1635765472340083 1.099045587774031; -0.883292194443226 1.68
58547637728394 … -2.6488571689716065 -0.885369657745364])
```

Problem 3

In [13]:
```
function predict(testdata,V,W)
    n = size(testdata,2)
    ar = []
    for j=1:n
        x = testdata[:,j]
        yxj = round.(Int64, σ.(W*(σ.(V*x))))
        r = mapstr[yxj[1]*2 + 1 + yxj[2]]
        push!(ar,r)
    end
    return permutedims(reshape(ar,(4,5)))
end
```

Out[13]:
```
predict (generic function with 1 method)
```

In [14]:
```
1 plot_chars(testdata)
2 predict(testdata, V, W)
```



Out[14]:
```
5×4 Matrix{Any}:
 'M'  'A'  'T'  'H'
 'A'  'H'  'A'  'T'
 'H'  'H'  'T'  'T'
 'A'  'H'  'H'  'T'
 'H'  'H'  'H'  'T'
```

Problem 4

In [16]:

```
using Optim

function train_optim(; maxiter=10000, rate=1)
    function objective(p)
        V = reshape(p[1:420], 10, 42)
        W = reshape(p[421:end], I'm 2, 10)
        total_loss = 0
        for j = 1:4, k = 1:2
            xj = training[:, j]
            Wk = W[k:k,:]
            r = σ.(V * xj)
            y = σ.(Wk r)[k]
            loss = (y - target[k, j])^2
            total_loss += loss
        end
        return total_loss
    end

    p0 = vcat(V[:], W[:])

    result = optimize(objective, p0, GradientDescent(), autodiff=:forw
                      iterations=maxiter, stepsize=rate)

    V = reshape(result.minimizer[1:420], 10, 42)
    W = reshape(result.minimizer[421:end], 2, 10)

    return V, W
end
```

```
syntax: missing comma or ) in argument list

Stacktrace:
 [1] top-level scope
   @ In[16]:6
 [2] eval
   @ ./boot.jl:360 [inlined]
 [3] include_string(mapexpr::typeof(REPL.softscope), mod::Module, code::S
tring, filename::String)
   @ Base ./loading.jl:1116
```

In [17]:

```
plot_chars(testdata)
V,W = train_optim()
predict(testdata, V, W)
```

```
UndefVarError: train_optim not defined

Stacktrace:
 [1] top-level scope
   @ In[17]:2
 [2] eval
   @ ./boot.jl:360 [inlined]
 [3] include_string(mapexpr::typeof(REPL.softscope), mod::Module, code::S
tring, filename::String)
   @ Base ./loading.jl:1116
```

In [ ]:

```

```