First we exist.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
//put together by Dominic Alexander Cooper
int main(){
        //Code Adapted by DAC from lyst on https://stackoverflow.com
        //k+1 = no. of elements
        //n = exponent = number of cells
        //the k and n values must perfectly fit the size of the set of elements in question
        FILE *p; p = fopen("SOLUTION_RENAME.txt","w");
        char a[] =
{'a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z','
','\n','\t','\\','\'',',','/','<','>','?',':',';','@','#','~',']','[','{','}','`','¬','|','¦','!','"','£','$','
%','^','&','*','(',')','-','_',' +','=','.','A','B','C','D','E','F','G','H',
'I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z','0','1','2','3','4','5','6','7','8','9'
};
        //char a[] = {'1','2','3','4','p1','p2','p3','p4','p5','p6'};
        //char b[] = {'a','b'};
        //char c[] = {'0','1','2','3'};
        //int n = 4; //int k = 3; //int n = x;
        //int k = 100;
        int k = strlen(a) - 1;
        printf("\n\tk = %d", k);
        int noc; printf("\n\tn = ");
        scanf("%d", &noc);
        printf("\n\tNumber Of FILE Cells = %d", noc);
        int n = noc;
        int row, col;
        int cell;
        int rdiv;
        int id;
        id = 0;
        int nbr_comb = pow(k+1, n);
        for (row=0; row < nbr_comb; row++){
                id++;  fprintf(p,"\n\nF%d\n\n", id);
                for (col=n-1; col>=0; col--){  rdiv = pow(k+1, col);
                        cell = (row/rdiv) % (k+1); fprintf(p,"%c", a[cell]);
                }
                printf("\n");
        }
        fprintf(p,"\n\nend.(k+1)^n = (%d + 1)^%d = %d", k, n, id);
        fclose(p);
        //end of adaptation
        return 0;
}
```

AND

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <iostream>
using namespace std;
//put together by Dominic Alexander Cooper
int main(){
        //Code Adapted by DAC from lyst on https://stackoverflow.com
        //k+1 = no. of elements
        //n = exponent = number of cells
        //the k and n values must perfectly fit the size of the set of elements in question
        FILE *p; p = fopen("SOLUTION_RENAME.txt","w");
        char a[] =
{'a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z','
','\n','\t','\\','\'',',','/','<','>','?',':',';','@','#','~',']','[','{','}','`','¬','|','¦','!','"','£','$','
%','^','&','*','(',')','-','_',' +','=','.','A','B','C','D','E','F','G','H',
'I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z','0','1','2','3','4','5','6','7','8','9'
};
        int pin;

        int pr, pc;
        printf("\nLet us begin\n\n");
        pc = 100;
        puts("26 = SPACE");
```

```
        puts("27 = NEWLINE");
        puts("28 = TAB");
        puts("");
        for(pr = 0; pr < pc; pr++){
                printf("%d      %c\n", pr, a[pr]);
        }

        cout << "Enter the size of your array: ";
        cin >> pin;
        int array[pin], inn, position;
        cout << "\n\n Enter the " << pin << " elements of your array: \n\n";
        for(inn = 0; inn < pin; inn++){
                cin >> array[inn];
        }
        //char a[] = {'1','2','3','4','p1','p2','p3','p4','p5','p6'};
        //char b[] = {'a','b'};
        //char c[] = {'0','1','2','3'};
        //int n = 4; //int k = 3; //int n = x;
        //int k = 100;

        //int k = strlen(a) - 1;
        int k;
        k = pin - 1;
        printf("\n\tk = %d", k);
        int noc; printf("\n\tn = ");
        scanf("%d", &noc);
        printf("\n\tNumber Of FILE Cells = %d", noc);
        int n = noc;
        int row, col;
        int cell;
        int rdiv;
        int id;
        id = 0;
        int rin;
        int nbr_comb = pow(k+1, n);
        for (row=0; row < nbr_comb; row++){
                id++;  fprintf(p,"\n\nF%d\n\n", id);
                for (col=n-1; col>=0; col--){  rdiv = pow(k+1, col);
                        //cell = (row/rdiv) % (k+1); fprintf(p,"%c", a[cell]);

                        cell = (row/rdiv) % (k+1);
                        rin = array[cell];
                        fprintf(p,"%c", a[rin]);
                }
                //printf("\n");
        }
        fprintf(p,"\n\nend.(k+1)^n = (%d + 1)^%d = %d", k, n, id);
        fclose(p);
        //end of adaptation
        return 0;
}
```

AND

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int main(){
        //FILE *p; p = fopen("SOLUTION.txt","w");
        char a[] =
{'a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z','
','\n','\t','\\','\'',',','/','<','>','?',':',';','@','#','~',']','[','{','}','`','¬','|','¦','!','"','£','$','
%','^','&','*','(',')','-','_',' +','=','.','A','B','C','D','E','F','G','H',
'I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z','0','1','2','3','4','5','6','7','8','9'
};
        int k, kprompt, fnum;
        k = sizeof(a) - 1;
        int soe = 0;
        puts("Set Of Elements");
        do{
                printf("\n\tElement %d = %c", soe, a[soe]);
                soe++;
        }while(soe <= k);
```

```
        printf("\n\tNumber Of Elements = %d", k + 1);
        printf("\n\tK = Number of Elements - 1");
        printf("\n\tTherefore k = %d", k);
        int noc, assert, go;
        printf("\n\tn = Number Of Cells Per File To Be Generated");
        printf("\n\n\t(k+1)^n = Number of Files To Be Generated");
        printf("\n\tEnter 2 = Search For A Specific File\n\tEnter 1 = Generate Files\n\tEnter 0 =
Abort\n\n\t"); scanf("%d", &go);
        if(go == 1){
                assert = 2;
        }else if(go == 2){
                assert = 1;
        }else if(go == 0){
                assert = 0;
        }else
                assert = 3;
        //Get Files From Database and save to directory
        if(assert == 2){
                printf("\n\tPlease Enter a Value For n:\t");
                scanf("%d", &noc); printf("\n\tNumber Of FILE Cells To Be Generated = %d", noc);
                int n = noc; int row, col; int cell; int rdiv; int id; id = 0;
                int nbr_comb = pow(k+1, n);
                for (row=0; row < nbr_comb; row++){
                        char filename[500];
                        sprintf(filename, "%dC%d.txt",n,id);
                        //FILE *p; p = fopen("SOLUTION.txt","w");
                        FILE *p; p = fopen(filename,"w");
                        id++;
                        //fprintf(p,"\n\n\n\nFILE%d\n\n\n\n", id);
                        for (col=n-1; col>=0; col--){
                                rdiv = pow(k+1, col); cell = (row/rdiv) % (k+1); fprintf(p,"%c", a[cell]);
                        }
                        if(id == nbr_comb){
                                fprintf(p,"\n\n\t(k+1)^n = (%d + 1)^%d = %d", k, n, id);
                        }
                        fclose(p);
                        printf("\n");
                }
        }
        //Get One Specific File From Database and save to active Directory
        if(assert == 1){

                printf("\n\tPlease Enter a Value For n:\t");
                scanf("%d", &noc);
                int n = noc;
                int nbr_comb = pow(k+1, n);
                printf("\n\n\tPlease Enter file number wanted between 0 and %d Inclusively:\t", nbr_comb-1);
                scanf("%d", &fnum);
                printf("\n\tNumber Of FILE Cells To Be Generated = %d", noc);
                int row, col; int cell; int rdiv; int id; id = fnum;

                int sw = 1;
                for (row=0; row < nbr_comb; row++){
                        if(row == fnum){
                                char filename[500];
                                sprintf(filename, "DQUERY%dC%d.txt",n,id);
                                //FILE *p; p = fopen("SOLUTION.txt","w");
                                FILE *p; p = fopen(filename,"w");
                                id++;
                                //fprintf(p,"\n\n\n\nFILE%d\n\n\n\n", id);
                                for (col=n-1; col>=0; col--){
                                        rdiv = pow(k+1, col); cell = (row/rdiv) % (k+1); fprintf(p,"%c",
a[cell]);
                                sw = 0;
                                }
                        if(sw == 0)
                                break;

                        }

                }


        }
```

```
        if(assert == 0 || assert == 3){
                puts("exiting");
        }
        return 0;
}


Then we must filter.

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <string.h>
#include <time.h>

int main(){


        //char i[4] = {""};
        int j;
        int tmax;
        puts("Note: Program Output --> FIdentified.txt --> In current Directory\n");
        printf("\nStatement/Word Character Length = ");
        scanf("%d", &tmax);
        j = tmax;
        printf("\n\t%d it is", j);
        char ui[tmax];
        printf("\n\nInput String (Max Length = 4): ");
        int x;
        puts("\n");
        scanf("%s", &ui);
        printf("Your Input:\n\t");
        for(x = 0; x < j; x++){
                printf("%c", ui[x]);
        }

        /*
        fn(v,p) = (v(a + i)(p(n- k)) + ... + v(0)) - (p(n - l) + ... + p(0));
        i = {1,2,3,4 ...};
        k = {1,2,3,4 ...};
        l = {1,2,3,4 ...};
        a = 0;
        p(n - k) = e^(n-k);
        e = number of potential events in the set of potential events;
        v(0) =  NULL;
        p(0) = NULL;
        (v1)(NULL) = v1;
        NULL + NULL = 0;
        n = number of events of the statement or of the word;
        Read from left to right;

        God

        --->

        (v1(p2) + v2(p1) + v3(p(0)) + v(0)) - (p2 + p1 + p(0))
        ((71*100^2) + (15*100^1) + (4*NULL) + NULL) - (100^2 + 100^1 + NULL);
        ((71*100^2) + (15*100^1) + (4)) - (100^2 + 100^1)
        701404
        3CF701404
        */
        char a[] =
{'a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z',' ','\n','\t','\\','\'',',','/','<','>','?',':',';','@','#','~',']','[','{','}','`','¬','|','¦','!','"','£','$','%','^','&','*','(',')','-','_',' +','=','.','A','B','C','D','E','F','G','H',
'I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z','0','1','2','3','4','5','6','7','8','9'
};

        //puts("How many characters long is your input: ");
        //scanf("%d", &j);
        int e,kmax,item,v,a1,i,p,n,k,l,pros,psum,image;
        long N;
```

```
        e = 100;
        a1 = 0;
        k = 1;
        n = j;
        kmax = n - k;
        int pr,f;
        f = 0;
        N = 0;
        if(n <= 4 && n > 0){
                        i = a1;
                        item = a1 + i;
                        while(f < n){
                                for(image = 0; image < e; image++){
                                        if(ui[item] == a[image]){
                                                N = N + ((image + 1)*(pow(e,n - k)));
                                                image = e;
                                                k++;
                                                item++;
                                        }

                                        i++;
                                }
                                f++;

                        }
                image = 0;
        }

        //N = N + v;
        int iter;
        for(iter = 0; iter < kmax; ++iter){
                N = N - (pow(e,kmax - iter));
        }

        FILE *pi;
        pi = fopen("FIdentified.txt","w");

        fprintf(pi,"%dCF%d\n\t",n,N);
        fprintf(pi,"%s", ui);
        printf("\n\nFile Identified:\n\t%dCF%d\n\n\t",n,N);

        fclose(pi);

        system("pause");

        return 0;
}
```

We then journey to the target state of space. By way of unitary purposeful movement.

```
from tkinter import *
import tkinter as tk
import random
import os
import math

import subprocess

import io #experiment
#import pyautogui
#from PIL import Image, experiment


print("random() : ", random.random())
master = Tk()
master.attributes('-fullscreen', True)
#a = 250
#b = 200
print("Welcome\n")
#print("\n\nPlease Ensure that the List of words begins with the horizontals then the verticals and that the
words are spelled only with lowercase letters. Thank you.")
print("Tip: side length should be a factor of the image width and of the image height")
change1 = input("Enter side length of image block:  ")
change = int(change1)
```

```
#a = 1880
#b = 1050
a1 = input("Enter width of image: ")
a = int(a1)
b1 = input ("Enter height of image:  ")
b = int(b1)
pin_p = a/change
w = Canvas(master, width= a, height= b)




#files = 3
#files_buffer = input("Enter number of files to be generated (8^((width/side_length)*(height/side_length)):  ")
#files = int(files_buffer)

#c = ["red","blue","yellow","brown","purple","pink","green","orange"]
c = ["white","black"]
ac =
["a","b","c","d","e","f","g","h","i","j","k","l","m","n","o","p","q","r","s","t","u","v","w","x","y","z","0"]
ba1 = []
ba2 = []
ba3 = [0,1]
ba4 = []
ba5 = []
word_lengths = []
word_letter_sums = []

xc = 0
zerox = 0
zeroy = 0
p = 1
range_for = int((a/change)*(b/change))
name = 1
cells = ((a//change)*(b//change))
upper = cells - 1
#nbr_comb = math.pow(len(c),cells)
#files = int(nbr_comb)
#print(len(c))
rown = 0


#rdiv = math.pow(len(c),cells - 1)
#print(rdiv)
#cell = (row/rdiv) % (len(c))
#print(cell)

#rdiv = math.pow(len(c),cells - 2)
#print(rdiv)
#cell = (row/rdiv) % (len(c))
#print(cell)
#def rone(h,j,l):
#    cell = (h/j) % l
#    celled = int(cell
#    w.create_rectangle(zerox, nyleft, nxright,nyright, fill = c[celled], outline = c[celled])
file_count = 0

#/////////////////////////////////////////
#Binding array Generator
#/////////////////////////////////////////


hine = open("WordList.txt","r")
eof = 0
counter1 = 0
pcells = ((a/b)/(change*change))
p_cells = int(pcells)
while(eof < 1):

    utah = hine.readline(1)
    if(utah != ''):
        ba1.append(utah)
```

```
    if(utah == '\n'):
        counter1 = counter1 + 1

    if(utah == ''):
        ba1.append("\n")
        eof = 1
print(ba1)
word_letter_summation = 0
letter_count = 0
counter2 = 0
hine.close()

while(counter2 < len(ba1)):
    if(ba1[counter2] != '\n'):
        for check1 in  range(0,len(ac)):
            if(ba1[counter2] == ac[check1]):
                ba4.append(check1 + 1)
                letter_count = letter_count + 1
    word_letter_summation = word_letter_summation + check1
    if(ba1[counter2] == '\n'):
        word_lengths.append(letter_count)
        word_letter_sums.append(word_letter_summation)
        word_letter_summation = 0
        letter_count = 0

    counter2 = counter2 + 1

print(word_lengths)
tisum = 0

for ti in range(0,len(word_lengths)):
    tisum = tisum + word_lengths[ti]
    tisum_ = int(tisum)
print(tisum_)

cells = (a*b)/(change*change)
counter3 = 0
while(counter3 < cells - (counter2 - 2)):
    ba4.append(0)
    counter3 = counter3 + 1

print(ba4)



#.........................................

#This is where the magic happens (sort ba4 into the binding array ba5)




#//////////////////////////////////////////


blockp_width = a/change
b_w = int(blockp_width)
blockp_height = b/change
b_h = int(blockp_height)

#horizontal word.... shift right, + 1 in pointer value
#vertical word..... shift down == + b_w .... (for a square crossword puzzle)

j1 = input("Number of files to be generated: ")
#j2 = input("Number of words per file: ")
j1_ = int(j1)
#j2_ = int(j2)
files = j1_
counter0 = 0
x = 0
prime = cells
prime_ = int(prime)
t = 0
yin = 0
```

```
#k1 = input("Number of horizontal words per file: ")
#k1_ = int(k1)
#k2 = input("Number of vertical words per file: ")
#k2_ = int(k2)

ih = 0


#pome = 0
calc = 0
#for inj in range(0,k1_):
#    pome = pome + word_lengths[inj]

#initialCell = input("Enter the number for the initial cell of your choice: ")
#initialCell_ = int(initialCell)

complete_ = 0
inu = 0
tabs = 0
while(complete_ != 1):
    tabs = tabs + 1

    if(tabs < prime_):
        while(calc < prime_):
            print("id", calc, "value", ba4[inu])
            take = input("Change or input the value of cell: ")
            if(take == 'c'):
                break
            take_ = int(take)
            ba4[calc] = take_
            calc = calc + 1
            inu = inu + 1
    complete = input("Complete [1, yes or 0, no] ")
    complete_ = int(complete)

    inu = 0
    calc = 0

while(t < files):
    file_count = file_count + 1
    #if(file_count == 121):
    #    break
    switch = 1
    sw = 0
    #for x in range(range_for):
    #for x in range (0,cells):
    #x = 0
    col = cells - 1
    #print(rown)
    while(x < prime_):

        #f = open('%s.ps' % name, 'wb')
        #f.close
        if(switch == 1):
            row = 1
            nxleft = 0
            nxright = change
            nyleft = 0
            nyright = change
            zerox = 0
            zeroy = 0
        c_length = len(c)
        switch = 0
        #ran = random.randint(0,c_length - 1)

        #rdiv = math.pow(len(c),col)
        #cell = (rown/rdiv) % (len(c))
        #rone(row,rdiv,len(c))
        #print(cell)
        #print(rdiv)
        #celled = int(cell)
        #print(celled)
        #print(cell)
```

```
        #w.create_rectangle(zerox, nyleft, nxright,nyright, fill = c[ran], outline = c[ran])
        #for puzzle_bind in range(0,):

        if(ba4[yin] != 0):
            w.create_rectangle(zerox, nyleft, nxright,nyright, fill = "white", outline = "black", width = 0)
        if(ba4[yin] == 0):
            w.create_rectangle(zerox, nyleft, nxright,nyright, fill = "black", outline = "black", width = 0)
        if(x <= cells):
            col = col - 1
        counter0 = counter0 + 1
        yin = yin + 1

        #w.create_rectangle(zerox, nyleft, nxright,nyright, fill = c[ran])
        #zxbuffer = zerox
        #w.place(x = zerox, y = zeroy)
        #w.place(bordermode = OUTSIDE, x = zerox + change, y = zeroy)
        w.grid(row = zeroy, column = zerox + change)
        if(p >= pin_p and p%pin_p == 0):
            zeroy = zeroy + change
            zerox = -change
            #zerox = 0
            nxleft = change
            nxright = 0
            nyleft = nyleft + change
            nyright = nyright + change
        zerox = zerox + change
        p = p + 1
        #xc + 1
        nxright = nxright + change
        if(xc == 3):
            xc = 0
        x = x + 1
    rown = rown + 1
    #counter0 = 0
    ce = str(name)
    w.update()
    w.postscript(file = ce + ".ps", colormode='color')
    name = name + 1
    if(x == prime_):
        x = 0
        if(yin == prime_*files):
            t = files
    #f_p = 'D:\\Kaliber\\Portfolio\\Content\\My PhotoBook\\Content\\'
    #os.rename(f_p + '1.ps', f_p + '2.ps')

    process = subprocess.Popen(["ps2pdf", ce + ".ps", ce + ".pdf"], shell=True)




    #Contender 1 For File Saving
    """
    def savefirst():
        cnv = getscreen().getcanvas()
        global hen
        ps = cnv.postscript(colormode = 'color')
        hen = filedialog.asksaveasfilename(defaultextension = '.jpg')
        im = Image.open(io.BytesIO(ps.encode('utf-8')))
        im.save(hen + '.jpg')
    #savefirst()
    """
    """
    Second Contender For File Saving
    def save(w):
        ps = w.canvas.postscript(colormode='color')
        img = Image.open(io.BytesIO(ps.encode('utf-8')))
        img.save('testing.jpg')
    """
#w.update()
#script  = ce + ".ps"
#w.save(script)
#os.startfile(script)
#print(script)
```

```
print("Done")


#master.mainloop()
```

Finally we maintain by moving within the bound of the desired state. That which governs this movement, is the Law. So, what is the Law? My thought, is that the Law is as follows: do what is best for the macrocosm.

Where the axioms of the stated Law are as follows:
- Only one mind is present.
- The role of the mind is to fulfill the determined will of the mind.
- Once true always true.
- The mind is the only structure that is present and complete.
- Division of the mind is not possible.
- Subtraction from the mind is not possible.
- Addition to the mind is not possible.
- Multiplication of the mind is not possible.
- The change of form of the mind is possible.
- No thing is negligible.
- The mind is composed of substructures.
- The movement of the substructures of the mind, enables the change of form of the mind.
- A word is a substructure of the mind.
- A fallacy is an undesirable substructure of the mind.