Peter Smith

Dr. Kahrs

ECE 1570

December 16, 2022

Final Exam

1. a) One number that can't be represented by the bfloat16 is $3.5 * 10^{38}$ simply because it maxes out the range of the float value. The advantages of bfloat16 is that it can store really large or small numbers just like a standard float while using half of the storage. What you gain in storage you lose in precision.

   b)

```
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
#include <algorithm>
#include <math.h>
#include <bitset>
#include <climits>
using namespace std;

unsigned short convert_float_to_bfloat(float f){
  union
  {
    float input;
    int   output;
  } data;
        data.input = f;
        bitset<sizeof(float) * CHAR_BIT> bits(data.output);
  bool a[16] ;
  int j = 0;
  for (int i = 31; i > 15; i--){
    a[j] = int(bits[i]);
    cout << a[j];
    j++;
  }

  cout <<endl<< bits << endl;
  unsigned short store = 0;
  for (int i = 0; i < 15; i++ , j--)
  store += pow(a[i]*2,j);
```

```
if (a[15] == 1)
   store += 1;
   cout << " bfloat is " << store << endl;
   return store;

}
float convert_bfloat_to_float(unsigned short bf){
   float store = 0;
   float a[16];
   for(int i=15; i>=0; i--)
{
a[i]=bf%2;
bf= bf/2;
}
for (int i = 0; i < 16; i++)
cout << a[i];
cout << endl;

unsigned short exp =  pow(2*a[1],7)+  pow(2*a[2],6)+ pow(2*a[3],5)+ pow(2*a[4],4)+
pow(2*a[5],3)+ pow(2*a[6],2)+ pow(2*a[7],1);
if ( a[8] == 1)
exp += 1;
cout << " exponent is " << exp << endl;
float mantissa = 1.0 + a[9]/2 + a[10]/4 + a[11]/8 + a[12]/16 + a[13]/32 + a[14]/64 + a[15]/128;
cout << " mantissa is " << mantissa << endl;
store = pow(-1,a[0]) * pow(2,exp-127) * mantissa;
return store;
}
unsigned short get_sin(unsigned short opp,unsigned short hyp){
   float opposite = convert_bfloat_to_float(opp);
   cout << "opposite is " << opposite << endl;
   float hypotenuse = convert_bfloat_to_float(hyp);
   cout << "hypotenuse is " << hypotenuse << endl;
   if (hypotenuse < opposite)
      return 0;
   float result = opposite / hypotenuse;
   cout << "result is" << result << endl;
   unsigned short store = convert_float_to_bfloat(result);
   return store;
}
unsigned short get_cos(unsigned short adj,unsigned short hyp){
   float adjacent = convert_bfloat_to_float(adj);
   cout << "adjacent is " << adjacent << endl;
   float hypotenuse = convert_bfloat_to_float(hyp);
   cout << "hypotenuse is " << hypotenuse << endl;
   if (hypotenuse < adjacent)
      return 0;
   float result = adjacent / hypotenuse;
```

```
    cout << "result is " << result << endl;
    unsigned short store = convert_float_to_bfloat(result);
    return store;
}
int main (){
    unsigned short a;
    unsigned short b;
    cout << "Enter value between 0 and 65535:" << endl;
    cin >> a;
    cout << "Enter value between 0 and 65535:" << endl;
    cin >> b;
    unsigned short sine = get_sin(a,b);
    cout << "sin result is " << sine << endl;
    unsigned short cos = get_cos(a,b);
    cout << "cos result is " << cos << endl;
    return 0;

};
```

2. A) With how the code is currently structured It lack cache coherence because processor C may be changing values stored in the other caches. Once these values are changed the calculations that were completed by Processor A and Processor B could be invalidated.

   B) To ensure better cache coherence I would change how these loops are being processed. I would run the loops in parallel. For example I would use openMP and split the loops between the three processors. Example:

```
#pragma omp parallel for
for  (int i=0, sum = 0; i< N;i++){
        if (FFT_input[i].real < 0) FFT_input[i].real = 0;}
#pragma omp parallel for
For (int I = 0, sum = 0; I < N; i++){
        Sum += cabs(FFT_input[i]);}
#pragma omp parallel for
For (int I = 0, sum = 0; I < N ; i++){
If (cabs(FFT_input[i])> max) max = FFT_input[i];}
```

3. A)

```
#include <stdlib.h>
#include <stdio.h>
#include <assert.h>
#include <vector>
#include <math.h>
#include <cmath>
#include <iostream>
#include <bits/stdc++.h>
#include <omp.h>
#include <time>
```

```
using namespace std;

int main()
{

        int N = 512;
        int image[N][N];

        srand((unsigned) time(NULL));

#pragma omp parallel for
        for (int i = 0; i < N*N ;i++){
                image[i/512][i%512] = (rand() %256);
        }

        for (int i = 0 ;i < N-1;i++){
                for (int j = 1; j < N - 1; j++){
                        #pragma oss task                                \
                        int old_pixel = image[i][j];
                        int new_pixel = trunc(image[i][j]+0.5);
      image[i][j] = new_pixel;
      int quant_error = old_pixel - new_pixel;                          \
     image[i][j+1] = image[i][j+1] + (quant_error * 7/16);              \
     image[i+1][j-1] = image[i+1][j-1] + (quant_error * 3/16);          \
     image[i+1][j] = image[i+1][j] + (quant_error * 5/16);              \
     image[i+1][j+1] = image[i+1][j+1] + (quant_error * 1/16);          \
                        }
                }
                return 0;
};
```

4. A)

```
#include <stdlib.h>
#include <cstdio>
#include <cstring>
#include <vector>
#include <math.h>
#include <cmath>
#include <iostream>
#include <fstream>
#include <complex>

using namespace std;
#define PI 3.14159265358

void split_signal(complex<double> *sig, int len){
        complex<double>* even_sig = new complex<double>[len/2];
        complex<double>* odd_sig = new complex<double>[len/2];
```

```
	int even_i = 0, odd_i = 0;
	for ( int i = 0; i < len; i++)
	{
		if ( ( i % 2) == 0){
			even_sig[even_i] = sig[i];
			even_i++;
		}
		else
		{
			odd_sig[odd_i] = sig[i];
			odd_i++;
		}
	}
	int size = len/2;
	memcpy(sig, even_sig, sizeof(complex<double>)*size);
	memcpy(sig+size, odd_sig, sizeof(complex<double>)*size);

	delete[] even_sig;
	delete[] odd_sig;
}
void fast_fourier_transform(complex<double> *sig, double N){
	if (N == 1)
	return;

	int split = N/2;
	complex<double> imag(0,1);
	complex<double> h_n = exp((2*PI*imag)/N);
	complex<double> h = 1;

	split_signal(sig,split*2);

	fast_fourier_transform(sig,split);
	fast_fourier_transform(sig+split,split);

	complex<double> even(0,0), odd(0,0);
	for (int i = 0; i < split;i++){
		even = sig[i];
		odd = sig[i+split];
		sig[i] = even + h*odd;
		sig[i+split] = even - h*odd;

		h = h*h_n;
	}
}
int main(int argc, char * argv[])
{
	int samples = 1024;
```

```
        ifstream infile;
        complex<double>* signal = new complex<double>[samples];
        infile.open("office.ascii");
        for (int i = 0; i < samples; i++)
                infile >> signal[i];
        infile.close();
        fast_fourier_transform(signal,samples);
        ofstream outfile;
        outfile.open("result.txt");
        outfile << "Fast Fourier Transform Results:" << endl;
        for(int i = 0; i < samples;i++)
        outfile << signal[i] << endl;

        outfile.close();

        delete[] signal;
};
```

Fast Fourier Transform Results:
(0.0003268,0)
(0.000513736,0.000285617)
(-0.00102481,0.00148281)
(0.00217965,-0.00611386)
(0.00457678,-0.00312766)
(0.00795045,0.00212169)
(0.00105608,0.0029152)
(-0.0041573,0.00420071)
(-0.00639289,0.00372646)
(-0.00904693,0.00398311)
(-0.00811984,-0.00208274)
(-0.00352936,-0.0057802)
(0.00110045,-0.00727227)
(0.000436609,-0.00630214)
(0.0082154,-0.00611166)
(0.0118147,0.00248627)
(0.00641746,0.00641548)
(0.00356802,0.00587584)
(-0.000890438,0.00618168)
(-0.000394463,0.00685849)
(-0.0049828,0.00489712)
(-0.00737744,-0.0014526)
(-0.00442742,-0.00504222)
(-0.00430955,-0.00109993)
(-0.00486489,-0.00978036)
(0.00713061,-0.00348795)
(0.00332861,-0.00561519)
(0.00440858,0.00296251)

(0.0025166,-0.00455668)
(0.0053079,0.00337067)
(0.00170067,0.000644856)
(0.000796828,0.00548449)
(-0.00125729,0.000633952)
(0.00379671,0.00271787)
(-0.00205428,0.00225106)
(-0.00955619,-9.57603e-005)
(-9.25665e-005,-0.00377054)
(-0.00167116,-0.00226449)
(-0.00256643,-0.00508622)
(-0.00321999,-0.00241857)
(-0.00193034,-0.00354527)
(-0.000201205,0.000102076)
(2.94991e-005,-0.000677613)
(0.00496,-0.00133527)
(0.00363907,-2.26428e-005)
(0.00494598,0.000323702)
(0.00188962,0.00220301)
(0.00158325,0.00517911)
(-0.000875002,0.00739917)
(-0.00708214,0.00714528)
(-0.00699838,-0.00142457)
(-0.00647135,-0.00225931)
(-0.00458995,-0.00772491)
(-0.00293977,-0.00689316)
(0.00229478,-0.0088047)
(0.00845988,-0.00299391)
(0.0087032,0.00105141)
(0.00835981,0.00749811)
(0.00382817,0.00862212)
(-0.00332036,0.0123073)
(-0.00929815,0.00378668)
(-0.0110269,-0.00224472)
(-0.00855773,-0.00952191)
(-0.00133087,-0.0126657)
(0.00800629,-0.00920269)
(0.0106874,-0.00359099)
(0.0114236,0.00494126)
(0.00416267,0.0127343)
(-0.000378362,0.0105488)
(-0.00475504,0.0110766)
(-0.0134367,0.00195888)
(-0.0122875,-0.00548769)
(-0.0126666,-0.0131673)
(-0.00382919,-0.0166464)
(0.00828136,-0.0180611)
(0.0198029,-0.0104815)

(0.017171,0.0118074)
(0.0068631,0.0164504)
(-0.00644799,0.0143422)
(-0.00891161,0.00800147)
(-0.0150711,0.00782034)
(-0.0103051,-0.00240001)
(-0.00534268,-0.000454584)
(-0.00411574,-0.00256397)
(-0.000406417,-0.00316751)
(0.00239059,-0.00571757)
(-0.000939145,-0.00540439)
(0.00807702,-0.00472892)
(0.00500605,-0.00580327)
(0.0100375,-0.0027998)
(0.0105665,0.00470778)
(0.0057242,0.00568547)
(0.00182427,0.0116074)
(-0.00140408,0.00986141)
(-0.00929766,0.0137244)
(-0.016176,0.00715211)
(-0.0105929,-0.00286034)
(-0.00735221,-0.0119603)
(-0.00883451,-0.00949752)
(0.000320843,-0.0162463)
(0.0131331,-0.0161141)
(0.0243502,-0.0128974)
(0.010214,0.00563619)
(0.0090698,0.00982563)
(0.00156621,0.0239007)
(-0.00913916,0.0137557)
(-0.0114225,0.00657232)
(-0.00809641,0.000311322)
(-0.00683525,-0.00416587)
(-0.00539181,-0.00644026)
(-0.000190695,-0.00888324)
(4.58496e-005,-0.00130974)
(-0.000225658,-0.00170758)
(0.000112002,-0.00129457)
(0.000856814,-0.00736516)
(0.00358401,-0.00297764)
(0.00586859,-0.001688)
(0.0063575,-0.000659443)
(0.00929657,0.00842159)
(0.00277168,0.00955069)
(0.00106418,0.00935254)
(-0.00323975,0.00623924)
(-0.00987113,0.00292023)
(-0.00868661,0.00179254)

(-0.0100375,-0.00315161)
(-0.0048672,-0.0070339)
(-2.65612e-005,-0.00775394)
(0.0047023,-0.00967303)
(0.00527361,-0.00616111)
(0.00626154,-0.00555058)
(0.00699543,0.000993225)
(0.00370562,0.00398107)
(0.00466798,0.00611532)
(0.00212584,0.00672243)
(-0.000538592,0.00821901)
(-0.00150206,0.00660513)
(-0.00561228,0.00556139)
(-0.0070443,0.00285102)
(-0.00922977,0.00119671)
(-0.0122467,-0.00403285)
(-0.00278593,-0.00995206)
(0.00141262,-0.0100352)
(0.0124508,-0.0126153)
(0.00751337,-0.00466483)
(0.0113589,-0.00245922)
(0.0077094,0.00789832)
(0.00445889,0.0101013)
(0.000927905,0.011807)
(-0.00163097,0.00595886)
(-0.00552708,0.00453802)
(-0.00764344,-6.88477e-005)
(-0.00792278,-0.00280902)
(-0.00497194,-0.00291652)
(-0.00450162,-0.00540805)
(-0.00121461,-0.00625922)
(0.00137545,-0.0083809)
(0.00400408,-0.00637928)
(0.00605136,-0.00051345)
(0.00391637,-0.00192486)
(0.00415517,0.000993879)
(0.00695478,0.0050507)
(0.00136132,0.00318729)
(-0.00166945,0.00753128)
(-0.000949158,0.00383387)
(-0.00678286,0.00882466)
(-0.00703491,0.00211414)
(-0.00647853,0.00240244)
(-0.00671403,-0.00375)
(-0.00113321,-0.00412716)
(-0.00190137,-0.0108709)
(0.00327251,-0.0124264)
(0.00769102,-0.00534971)

(0.00222642,-0.000840956)
(0.00743419,0.00534373)
(0.00417313,0.00519154)
(0.00949508,0.00509183)
(-0.00253687,0.0046204)
(-0.00325975,0.00304372)
(-0.00469702,0.0116499)
(-0.00621214,0.00511419)
(-0.00678377,0.00195531)
(-0.00585958,-0.00368855)
(-0.0100929,-0.00935129)
(-0.00408465,-0.00424357)
(0.00232143,-0.0103686)
(0.0132787,-0.00751747)
(0.0106081,-0.00396228)
(0.0114198,-0.00242753)
(0.00730027,0.00860709)
(0.00501697,0.00836179)
(-0.002084,0.0114439)
(-0.00723684,0.00413511)
(-0.00753116,0.00632264)
(-0.00967366,2.65901e-005)
(-0.0067222,-0.002737)
(-0.00746739,-0.00504292)
(-0.00331824,-0.00835816)
(0.000330339,-0.0067898)
(0.00356964,-0.00947161)
(0.00842023,-0.00572228)
(0.0085236,-0.00222862)
(0.00746602,0.0049124)
(0.00352373,0.00906198)
(0.00496613,0.00926286)
(-0.00217585,0.00970436)
(-0.00729186,0.00658904)
(-0.0101684,0.00394836)
(-0.00432288,-0.00560968)
(-0.0103595,-0.00478185)
(-0.00251049,-0.00613384)
(0.00151135,-0.00514194)
(0.00601079,-0.00660475)
(0.00461532,-0.00342646)
(0.00734523,-0.000454357)
(0.00393442,0.00257554)
(0.00210632,0.00227505)
(-0.000998914,0.00597917)
(-0.000217061,0.00515222)
(-0.00225644,0.00476395)
(-0.00197783,0.00248897)

(-0.00359663,-8.17574e-006)
(-0.00227888,-0.00277086)
(-0.0046453,-0.00197603)
(-0.000813627,-0.00260444)
(-0.00169639,-0.00104777)
(0.00190368,-0.00239891)
(0.00283747,-0.00124851)
(0.00181835,-0.00175533)
(0.0026898,6.2953e-005)
(0.00173389,0.00135166)
(0.00176287,0.00312025)
(0.000993276,0.00314289)
(0.000527779,0.00215586)
(-0.000599633,0.00224845)
(-0.00206166,0.000635593)
(-0.00224489,0.000416179)
(-0.00215817,-0.000242358)
(-0.000824924,-0.000691921)
(-0.00122197,-0.00134042)
(-0.000167727,-0.00234079)
(0.000662926,-0.00167657)
(0.00127272,-0.00138296)
(0.00140352,-0.000927524)
(0.00232089,8.72969e-005)
(0.00193008,0.00103807)
(0.00121197,0.00136788)
(0.000594668,0.00163881)
(-4.26202e-005,0.00179109)
(-0.000463038,0.00171582)
(-0.00141562,0.00048846)
(-0.000951146,0.000514391)
(-0.00188701,-0.000358433)
(-0.000960153,-0.000488111)
(-0.00153839,-0.00100989)
(-0.000290967,-0.00166581)
(-0.000596361,-0.000766149)
(0.000466004,-0.0014306)
(0.00171304,-0.000834319)
(0.00195459,-0.000316952)
(0.00244304,0.000427031)
(0.00129705,0.000758514)
(0.00097511,0.00192077)
(5.88445e-005,0.00234636)
(-0.000670871,0.00193494)
(-0.00134425,0.00109282)
(-0.00179748,0.000502907)
(-0.00215122,-0.000662502)
(-0.00135126,-0.000284634)

(-0.00045494,-0.00149585)
(0.000366544,-0.00130439)
(0.000509071,-0.00152176)
(0.00174741,-0.00181087)
(0.00109731,-0.00079102)
(0.00145501,6.38469e-005)
(0.00168913,0.00154873)
(0.000870048,0.00115369)
(0.00092277,0.00182281)
(-0.000387905,0.0015098)
(-0.000920505,0.00092674)
(-0.00138895,0.00108278)
(-0.00166578,-0.000202168)
(-0.00110796,-0.000233696)
(-0.000941265,-0.00141412)
(0.000254915,-0.00145915)
(-9.16302e-005,-0.00111363)
(0.000648196,-0.00119794)
(0.00068246,-0.000406974)
(0.00122439,0.000101286)
(0.00124697,0.000612606)
(0.00057493,5.59559e-005)
(0.000410216,0.000544404)
(0.000445343,0.000468029)
(-0.000458252,0.00123722)
(0.000160985,0.00118304)
(-0.000202743,0.000953455)
(-0.000320136,8.53193e-005)
(-0.00116861,0.000553697)
(-0.00157254,-0.00108885)
(-0.00119218,0.000458126)
(-0.00141394,-0.000491308)
(1.36202e-005,-0.00088973)
(-0.00033507,-0.00155497)
(0.00111943,-0.00167955)
(0.00138737,-0.00116988)
(0.00177886,-0.000871817)
(0.00274293,0.000735029)
(0.00211501,0.00166202)
(0.00144577,0.00308681)
(0.000470052,0.0033214)
(-0.00162404,0.0024322)
(-0.00309073,0.00268533)
(-0.00433849,3.18992e-005)
(-0.00393058,-0.00200589)
(-0.00235487,-0.00310197)
(-0.000204098,-0.00414972)
(0.00186445,-0.00334988)

(0.00434176,-0.00248752)
(0.00507318,-9.10238e-005)
(0.00373712,0.000965315)
(0.00294139,0.00311277)
(0.000749651,0.00386908)
(-0.00108177,0.00448122)
(-0.002651,0.00342599)
(-0.00326454,0.00119041)
(-0.00322053,-0.000247194)
(-0.00262157,-0.00231573)
(-0.00180017,-0.00351515)
(-0.000665711,-0.00319173)
(0.000661413,-0.00244761)
(0.00234758,-0.00223736)
(0.00254523,-0.00157454)
(0.0039049,-0.000423323)
(0.00327653,0.00134791)
(0.00189316,0.00209459)
(0.00155575,0.00326234)
(-0.000109144,0.00333456)
(-0.00185986,0.00355017)
(-0.0033921,0.00203192)
(-0.0037199,0.00105737)
(-0.00443242,-0.000127777)
(-0.00399192,-0.00340501)
(-0.00206667,-0.00458952)
(0.000911553,-0.00674088)
(0.00382616,-0.00482532)
(0.00644612,-0.00221621)
(0.00757273,0.00147216)
(0.00546504,0.00550865)
(0.00119197,0.00665123)
(-0.0025367,0.0063645)
(-0.00478093,0.00356098)
(-0.00636054,0.00113949)
(-0.00542183,-0.00212104)
(-0.00340365,-0.00433141)
(-0.000466068,-0.00420458)
(0.00147027,-0.00459273)
(0.00335273,-0.00315435)
(0.00406156,-0.00169862)
(0.00421876,0.00103416)
(0.00265316,0.00304017)
(0.00155525,0.00312885)
(-7.26438e-005,0.00365621)
(-0.0014906,0.00270452)
(-0.00258253,0.00127072)
(-0.00288982,-0.000544402)

(-0.0024542,-0.00121137)
(-0.00152602,-0.00152611)
(-0.000462737,-0.00221225)
(0.00125477,-0.00178455)
(0.00139564,-0.000886573)
(0.00163739,-0.000617831)
(0.00168733,2.44703e-005)
(0.00133868,0.000491682)
(0.000644299,0.00105076)
(-3.05777e-005,0.000914257)
(0.000136736,0.00112156)
(-0.000505237,0.0010115)
(-0.000752997,0.00041426)
(-0.000929372,0.00027389)
(-0.000766052,-0.000368328)
(-0.000594736,-0.000392304)
(-0.000888461,-0.000895422)
(8.32387e-005,-0.000778533)
(7.82574e-005,-0.00058083)
(0.000924777,-0.000526554)
(0.000679796,-0.000181614)
(0.00132759,-0.000215012)
(0.000702401,0.000680256)
(0.000461145,0.000642768)
(-9.40738e-005,0.000842652)
(-0.000425788,0.00064048)
(-0.00078937,0.000414484)
(-0.000500365,-5.72865e-005)
(-0.000471183,-0.000308776)
(-0.000108179,-0.000300959)
(-0.000121397,-0.000351507)
(0.00036796,-0.000352943)
(0.000213967,2.59277e-005)
(0.000260623,-9.85896e-005)
(0.0001684,9.06714e-005)
(0.000200446,-4.44617e-005)
(8.24853e-005,0.000202646)
(9.52055e-005,-3.58675e-005)
(0.000138526,0.000123104)
(0.000155243,7.26822e-005)
(0.00016066,9.69949e-005)
(0.000125258,0.000133364)
(7.23036e-005,0.00023148)
(-2.42686e-005,0.000293896)
(-0.000215407,8.03436e-005)
(-0.000135464,-1.42628e-005)
(-0.000186797,-0.000121114)
(-6.24083e-005,-0.00014847)

(3.91462e-005,-0.000232951)
(0.000204306,-7.59869e-005)
(4.76881e-005,-3.9938e-005)
(0.000121985,-5.01426e-005)
(0.0001444,-0.000139889)
(0.000290307,-7.0816e-005)
(0.000139526,-2.95876e-005)
(0.000350588,9.19971e-006)
(0.000330514,0.000251162)
(0.000181197,0.000423537)
(-2.92443e-005,0.000327965)
(-0.000123454,0.000293597)
(-0.000394882,0.00024847)
(-0.000458223,2.931e-005)
(-0.000335871,-0.000282917)
(-0.000145302,-0.000433179)
(-4.5693e-005,-0.000488473)
(0.000161982,-0.000519608)
(0.000533996,-0.000410868)
(0.000615986,3.56845e-005)
(0.000527804,0.000267704)
(0.000446176,0.000378558)
(0.000271701,0.000562572)
(-9.84068e-005,0.000617137)
(-0.000304841,0.000396929)
(-0.000435481,0.000268138)
(-0.000449031,-2.49461e-005)
(-0.000445488,-0.000289781)
(-0.000316274,-0.000517513)
(4.30219e-005,-0.000514037)
(0.000183831,-0.000413426)
(0.000435449,-0.000262707)
(0.000689371,-3.64509e-005)
(0.000567619,0.000232112)
(0.000413818,0.000258152)
(0.000210925,0.000442305)
(3.30329e-005,0.000384128)
(-0.000257433,0.00038066)
(-0.000251469,0.000180647)
(-0.00026559,7.62717e-005)
(-0.000226575,-0.000135643)
(-0.000167751,-0.000226516)
(-0.000129605,-0.000336002)
(3.53975e-005,-0.000339913)
(0.00014217,-0.000297429)
(0.000322263,-0.000229963)
(0.000366428,-7.39308e-005)
(0.000409672,6.24949e-005)

(0.000305823,0.000205096)
(0.00020426,0.000275904)
(3.60722e-005,0.00033928)
(-5.07221e-005,0.000255101)
(-0.000188194,0.000254236)
(-0.000250975,7.66373e-005)
(-0.000267943,4.41731e-005)
(-0.000221019,-0.000159273)
(-0.000147949,-0.00025657)
(1.12053e-005,-0.00038295)
(0.000149005,-0.000302781)
(0.000305961,-0.000288295)
(0.00040722,-6.84311e-005)
(0.000428447,2.91674e-005)
(0.000406868,0.000285625)
(0.000179011,0.000342958)
(3.35945e-006,0.000445803)
(-0.000231639,0.000302681)
(-0.000296814,0.000123133)
(-0.000311288,-8.18987e-005)
(-0.000191246,-0.000197712)
(-5.21121e-005,-0.000276378)
(7.90465e-005,-0.000276103)
(0.000168128,-0.000190712)
(0.000237228,-0.00011867)
(0.000207919,-1.07898e-005)
(0.000221676,-1.76672e-005)
(0.000230707,5.10239e-005)
(0.000177981,0.000164652)
(6.42301e-005,0.000188055)
(-2.59178e-005,0.000228937)
(-9.3379e-005,0.000222644)
(-0.000147493,7.59995e-005)
(-0.000169355,-3.44451e-005)
(-0.000140903,-0.000170892)
(2.74412e-005,-0.000217131)
(0.000137138,-0.000198237)
(0.000209455,-6.50821e-005)
(0.000171861,-3.72464e-006)
(0.000164088,5.45943e-005)
(9.66083e-005,9.25167e-005)
(5.05903e-005,6.69064e-005)
(2.95797e-005,7.83488e-005)
(3.68224e-006,4.25253e-005)
(-9.79078e-006,3.57378e-005)
(-8.69599e-006,-2.30243e-005)
(5.19126e-005,-2.46742e-005)
(5.24407e-005,-3.13907e-005)

(6.76761e-005,-1.15862e-005)
(5.31686e-005,-8.38594e-006)
(7.88103e-005,8.98945e-006)
(4.23242e-005,1.39221e-005)
(5.83759e-005,0)
(4.23242e-005,-1.39221e-005)
(7.88103e-005,-8.98945e-006)
(5.31686e-005,8.38594e-006)
(6.76761e-005,1.15862e-005)
(5.24407e-005,3.13907e-005)
(5.19126e-005,2.46742e-005)
(-8.69599e-006,2.30243e-005)
(-9.79078e-006,-3.57378e-005)
(3.68224e-006,-4.25253e-005)
(2.95797e-005,-7.83488e-005)
(5.05903e-005,-6.69064e-005)
(9.66083e-005,-9.25167e-005)
(0.000164088,-5.45943e-005)
(0.000171861,3.72464e-006)
(0.000209455,6.50821e-005)
(0.000137138,0.000198237)
(2.74412e-005,0.000217131)
(-0.000140903,0.000170892)
(-0.000169355,3.44451e-005)
(-0.000147493,-7.59995e-005)
(-9.3379e-005,-0.000222644)
(-2.59178e-005,-0.000228937)
(6.42301e-005,-0.000188055)
(0.000177981,-0.000164652)
(0.000230707,-5.10239e-005)
(0.000221676,1.76672e-005)
(0.000207919,1.07898e-005)
(0.000237228,0.00011867)
(0.000168128,0.000190712)
(7.90465e-005,0.000276103)
(-5.21121e-005,0.000276378)
(-0.000191246,0.000197712)
(-0.000311288,8.18987e-005)
(-0.000296814,-0.000123133)
(-0.000231639,-0.000302681)
(3.35945e-006,-0.000445803)
(0.000179011,-0.000342958)
(0.000406868,-0.000285625)
(0.000428447,-2.91674e-005)
(0.00040722,6.84311e-005)
(0.000305961,0.000288295)
(0.000149005,0.000302781)
(1.12053e-005,0.00038295)

(-0.000147949,0.00025657)
(-0.000221019,0.000159273)
(-0.000267943,-4.41731e-005)
(-0.000250975,-7.66373e-005)
(-0.000188194,-0.000254236)
(-5.07221e-005,-0.000255101)
(3.60722e-005,-0.00033928)
(0.00020426,-0.000275904)
(0.000305823,-0.000205096)
(0.000409672,-6.24949e-005)
(0.000366428,7.39308e-005)
(0.000322263,0.000229963)
(0.00014217,0.000297429)
(3.53975e-005,0.000339913)
(-0.000129605,0.000336002)
(-0.000167751,0.000226516)
(-0.000226575,0.000135643)
(-0.00026559,-7.62717e-005)
(-0.000251469,-0.000180647)
(-0.000257433,-0.00038066)
(3.30329e-005,-0.000384128)
(0.000210925,-0.000442305)
(0.000413818,-0.000258152)
(0.000567619,-0.000232112)
(0.000689371,3.64509e-005)
(0.000435449,0.000262707)
(0.000183831,0.000413426)
(4.30219e-005,0.000514037)
(-0.000316274,0.000517513)
(-0.000445488,0.000289781)
(-0.000449031,2.49461e-005)
(-0.000435481,-0.000268138)
(-0.000304841,-0.000396929)
(-9.84068e-005,-0.000617137)
(0.000271701,-0.000562572)
(0.000446176,-0.000378558)
(0.000527804,-0.000267704)
(0.000615986,-3.56845e-005)
(0.000533996,0.000410868)
(0.000161982,0.000519608)
(-4.5693e-005,0.000488473)
(-0.000145302,0.000433179)
(-0.000335871,0.000282917)
(-0.000458223,-2.931e-005)
(-0.000394882,-0.00024847)
(-0.000123454,-0.000293597)
(-2.92443e-005,-0.000327965)
(0.000181197,-0.000423537)

(0.000330514,-0.000251162)
(0.000350588,-9.19971e-006)
(0.000139526,2.95876e-005)
(0.000290307,7.0816e-005)
(0.0001444,0.000139889)
(0.000121985,5.01426e-005)
(4.76881e-005,3.9938e-005)
(0.000204306,7.59869e-005)
(3.91462e-005,0.000232951)
(-6.24083e-005,0.00014847)
(-0.000186797,0.000121114)
(-0.000135464,1.42628e-005)
(-0.000215407,-8.03436e-005)
(-2.42686e-005,-0.000293896)
(7.23036e-005,-0.00023148)
(0.000125258,-0.000133364)
(0.00016066,-9.69949e-005)
(0.000155243,-7.26822e-005)
(0.000138526,-0.000123104)
(9.52055e-005,3.58675e-005)
(8.24853e-005,-0.000202646)
(0.000200446,4.44617e-005)
(0.0001684,-9.06714e-005)
(0.000260623,9.85896e-005)
(0.000213967,-2.59277e-005)
(0.00036796,0.000352943)
(-0.000121397,0.000351507)
(-0.000108179,0.000300959)
(-0.000471183,0.000308776)
(-0.000500365,5.72865e-005)
(-0.00078937,-0.000414484)
(-0.000425788,-0.00064048)
(-9.40738e-005,-0.000842652)
(0.000461145,-0.000642768)
(0.000702401,-0.000680256)
(0.00132759,0.000215012)
(0.000679796,0.000181614)
(0.000924777,0.000526554)
(7.82574e-005,0.00058083)
(8.32387e-005,0.000778533)
(-0.000888461,0.000895422)
(-0.000594736,0.000392304)
(-0.000766052,0.000368328)
(-0.000929372,-0.00027389)
(-0.000752997,-0.00041426)
(-0.000505237,-0.0010115)
(0.000136736,-0.00112156)
(-3.05777e-005,-0.000914257)

(0.000644299,-0.00105076)
(0.00133868,-0.000491682)
(0.00168733,-2.44703e-005)
(0.00163739,0.000617831)
(0.00139564,0.000886573)
(0.00125477,0.00178455)
(-0.000462737,0.00221225)
(-0.00152602,0.00152611)
(-0.0024542,0.00121137)
(-0.00288982,0.000544402)
(-0.00258253,-0.00127072)
(-0.0014906,-0.00270452)
(-7.26438e-005,-0.00365621)
(0.00155525,-0.00312885)
(0.00265316,-0.00304017)
(0.00421876,-0.00103416)
(0.00406156,0.00169862)
(0.00335273,0.00315435)
(0.00147027,0.00459273)
(-0.000466068,0.00420458)
(-0.00340365,0.00433141)
(-0.00542183,0.00212104)
(-0.00636054,-0.00113949)
(-0.00478093,-0.00356098)
(-0.0025367,-0.0063645)
(0.00119197,-0.00665123)
(0.00546504,-0.00550865)
(0.00757273,-0.00147216)
(0.00644612,0.00221621)
(0.00382616,0.00482532)
(0.000911553,0.00674088)
(-0.00206667,0.00458952)
(-0.00399192,0.00340501)
(-0.00443242,0.000127777)
(-0.0037199,-0.00105737)
(-0.0033921,-0.00203192)
(-0.00185986,-0.00355017)
(-0.000109144,-0.00333456)
(0.00155575,-0.00326234)
(0.00189316,-0.00209459)
(0.00327653,-0.00134791)
(0.0039049,0.000423323)
(0.00254523,0.00157454)
(0.00234758,0.00223736)
(0.000661413,0.00244761)
(-0.000665711,0.00319173)
(-0.00180017,0.00351515)
(-0.00262157,0.00231573)

(-0.00322053,0.000247194)
(-0.00326454,-0.00119041)
(-0.002651,-0.00342599)
(-0.00108177,-0.00448122)
(0.000749651,-0.00386908)
(0.00294139,-0.00311277)
(0.00373712,-0.000965315)
(0.00507318,9.10238e-005)
(0.00434176,0.00248752)
(0.00186445,0.00334988)
(-0.000204098,0.00414972)
(-0.00235487,0.00310197)
(-0.00393058,0.00200589)
(-0.00433849,-3.18992e-005)
(-0.00309073,-0.00268533)
(-0.00162404,-0.0024322)
(0.000470052,-0.0033214)
(0.00144577,-0.00308681)
(0.00211501,-0.00166202)
(0.00274293,-0.000735029)
(0.00177886,0.000871817)
(0.00138737,0.00116988)
(0.00111943,0.00167955)
(-0.00033507,0.00155497)
(1.36202e-005,0.00088973)
(-0.00141394,0.000491308)
(-0.00119218,-0.000458126)
(-0.00157254,0.00108885)
(-0.00116861,-0.000553697)
(-0.000320136,-8.53193e-005)
(-0.000202743,-0.000953455)
(0.000160985,-0.00118304)
(-0.000458252,-0.00123722)
(0.000445343,-0.000468029)
(0.000410216,-0.000544404)
(0.00057493,-5.59559e-005)
(0.00124697,-0.000612606)
(0.00122439,-0.000101286)
(0.00068246,0.000406974)
(0.000648196,0.00119794)
(-9.16302e-005,0.00111363)
(0.000254915,0.00145915)
(-0.000941265,0.00141412)
(-0.00110796,0.000233696)
(-0.00166578,0.000202168)
(-0.00138895,-0.00108278)
(-0.000920505,-0.00092674)
(-0.000387905,-0.0015098)

(0.00092277,-0.00182281)
(0.000870048,-0.00115369)
(0.00168913,-0.00154873)
(0.00145501,-6.38469e-005)
(0.00109731,0.00079102)
(0.00174741,0.00181087)
(0.000509071,0.00152176)
(0.000366544,0.00130439)
(-0.00045494,0.00149585)
(-0.00135126,0.000284634)
(-0.00215122,0.000662502)
(-0.00179748,-0.000502907)
(-0.00134425,-0.00109282)
(-0.000670871,-0.00193494)
(5.88445e-005,-0.00234636)
(0.00097511,-0.00192077)
(0.00129705,-0.000758514)
(0.00244304,-0.000427031)
(0.00195459,0.000316952)
(0.00171304,0.000834319)
(0.000466004,0.0014306)
(-0.000596361,0.000766149)
(-0.000290967,0.00166581)
(-0.00153839,0.00100989)
(-0.000960153,0.000488111)
(-0.00188701,0.000358433)
(-0.000951146,-0.000514391)
(-0.00141562,-0.00048846)
(-0.000463038,-0.00171582)
(-4.26202e-005,-0.00179109)
(0.000594668,-0.00163881)
(0.00121197,-0.00136788)
(0.00193008,-0.00103807)
(0.00232089,-8.72969e-005)
(0.00140352,0.000927524)
(0.00127272,0.00138296)
(0.000662926,0.00167657)
(-0.000167727,0.00234079)
(-0.00122197,0.00134042)
(-0.000824924,0.000691921)
(-0.00215817,0.000242358)
(-0.00224489,-0.000416179)
(-0.00206166,-0.000635593)
(-0.000599633,-0.00224845)
(0.000527779,-0.00215586)
(0.000993276,-0.00314289)
(0.00176287,-0.00312025)
(0.00173389,-0.00135166)

(0.0026898,-6.2953e-005)
(0.00181835,0.00175533)
(0.00283747,0.00124851)
(0.00190368,0.00239891)
(-0.00169639,0.00104777)
(-0.000813627,0.00260444)
(-0.0046453,0.00197603)
(-0.00227888,0.00277086)
(-0.00359663,8.17574e-006)
(-0.00197783,-0.00248897)
(-0.00225644,-0.00476395)
(-0.000217061,-0.00515222)
(-0.000998914,-0.00597917)
(0.00210632,-0.00227505)
(0.00393442,-0.00257554)
(0.00734523,0.000454357)
(0.00461532,0.00342646)
(0.00601079,0.00660475)
(0.00151135,0.00514194)
(-0.00251049,0.00613384)
(-0.0103595,0.00478185)
(-0.00432288,0.00560968)
(-0.0101684,-0.00394836)
(-0.00729186,-0.00658904)
(-0.00217585,-0.00970436)
(0.00496613,-0.00926286)
(0.00352373,-0.00906198)
(0.00746602,-0.0049124)
(0.0085236,0.00222862)
(0.00842023,0.00572228)
(0.00356964,0.00947161)
(0.000330339,0.0067898)
(-0.00331824,0.00835816)
(-0.00746739,0.00504292)
(-0.0067222,0.002737)
(-0.00967366,-2.65901e-005)
(-0.00753116,-0.00632264)
(-0.00723684,-0.00413511)
(-0.002084,-0.0114439)
(0.00501697,-0.00836179)
(0.00730027,-0.00860709)
(0.0114198,0.00242753)
(0.0106081,0.00396228)
(0.0132787,0.00751747)
(0.00232143,0.0103686)
(-0.00408465,0.00424357)
(-0.0100929,0.00935129)
(-0.00585958,0.00368855)

(-0.00678377,-0.00195531)
(-0.00621214,-0.00511419)
(-0.00469702,-0.0116499)
(-0.00325975,-0.00304372)
(-0.00253687,-0.0046204)
(0.00949508,-0.00509183)
(0.00417313,-0.00519154)
(0.00743419,-0.00534373)
(0.00222642,0.000840956)
(0.00769102,0.00534971)
(0.00327251,0.0124264)
(-0.00190137,0.0108709)
(-0.00113321,0.00412716)
(-0.00671403,0.00375)
(-0.00647853,-0.00240244)
(-0.00703491,-0.00211414)
(-0.00678286,-0.00882466)
(-0.000949158,-0.00383387)
(-0.00166945,-0.00753128)
(0.00136132,-0.00318729)
(0.00695478,-0.0050507)
(0.00415517,-0.000993879)
(0.00391637,0.00192486)
(0.00605136,0.00051345)
(0.00400408,0.00637928)
(0.00137545,0.0083809)
(-0.00121461,0.00625922)
(-0.00450162,0.00540805)
(-0.00497194,0.00291652)
(-0.00792278,0.00280902)
(-0.00764344,6.88477e-005)
(-0.00552708,-0.00453802)
(-0.00163097,-0.00595886)
(0.000927905,-0.011807)
(0.00445889,-0.0101013)
(0.0077094,-0.00789832)
(0.0113589,0.00245922)
(0.00751337,0.00466483)
(0.0124508,0.0126153)
(0.00141262,0.0100352)
(-0.00278593,0.00995206)
(-0.0122467,0.00403285)
(-0.00922977,-0.00119671)
(-0.0070443,-0.00285102)
(-0.00561228,-0.00556139)
(-0.00150206,-0.00660513)
(-0.000538592,-0.00821901)
(0.00212584,-0.00672243)

(0.00466798,-0.00611532)
(0.00370562,-0.00398107)
(0.00699543,-0.000993225)
(0.00626154,0.00555058)
(0.00527361,0.00616111)
(0.0047023,0.00967303)
(-2.65612e-005,0.00775394)
(-0.0048672,0.0070339)
(-0.0100375,0.00315161)
(-0.00868661,-0.00179254)
(-0.00987113,-0.00292023)
(-0.00323975,-0.00623924)
(0.00106418,-0.00935254)
(0.00277168,-0.00955069)
(0.00929657,-0.00842159)
(0.0063575,0.000659443)
(0.00586859,0.001688)
(0.00358401,0.00297764)
(0.000856814,0.00736516)
(0.000112002,0.00129457)
(-0.000225658,0.00170758)
(4.58496e-005,0.00130974)
(-0.000190695,0.00888324)
(-0.00539181,0.00644026)
(-0.00683525,0.00416587)
(-0.00809641,-0.000311322)
(-0.0114225,-0.00657232)
(-0.00913916,-0.0137557)
(0.00156621,-0.0239007)
(0.0090698,-0.00982563)
(0.010214,-0.00563619)
(0.0243502,0.0128974)
(0.0131331,0.0161141)
(0.000320843,0.0162463)
(-0.00883451,0.00949752)
(-0.00735221,0.0119603)
(-0.0105929,0.00286034)
(-0.016176,-0.00715211)
(-0.00929766,-0.0137244)
(-0.00140408,-0.00986141)
(0.00182427,-0.0116074)
(0.0057242,-0.00568547)
(0.0105665,-0.00470778)
(0.0100375,0.0027998)
(0.00500605,0.00580327)
(0.00807702,0.00472892)
(-0.000939145,0.00540439)
(0.00239059,0.00571757)

(-0.000406417,0.00316751)
(-0.00411574,0.00256397)
(-0.00534268,0.000454584)
(-0.0103051,0.00240001)
(-0.0150711,-0.00782034)
(-0.00891161,-0.00800147)
(-0.00644799,-0.0143422)
(0.0068631,-0.0164504)
(0.017171,-0.0118074)
(0.0198029,0.0104815)
(0.00828136,0.0180611)
(-0.00382919,0.0166464)
(-0.0126666,0.0131673)
(-0.0122875,0.00548769)
(-0.0134367,-0.00195888)
(-0.00475504,-0.0110766)
(-0.000378362,-0.0105488)
(0.00416267,-0.0127343)
(0.0114236,-0.00494126)
(0.0106874,0.00359099)
(0.00800629,0.00920269)
(-0.00133087,0.0126657)
(-0.00855773,0.00952191)
(-0.0110269,0.00224472)
(-0.00929815,-0.00378668)
(-0.00332036,-0.0123073)
(0.00382817,-0.00862212)
(0.00835981,-0.00749811)
(0.0087032,-0.00105141)
(0.00845988,0.00299391)
(0.00229478,0.0088047)
(-0.00293977,0.00689316)
(-0.00458995,0.00772491)
(-0.00647135,0.00225931)
(-0.00699838,0.00142457)
(-0.00708214,-0.00714528)
(-0.000875002,-0.00739917)
(0.00158325,-0.00517911)
(0.00188962,-0.00220301)
(0.00494598,-0.000323702)
(0.00363907,2.26428e-005)
(0.00496,0.00133527)
(2.94991e-005,0.000677613)
(-0.000201205,-0.000102076)
(-0.00193034,0.00354527)
(-0.00321999,0.00241857)
(-0.00256643,0.00508622)
(-0.00167116,0.00226449)

(-9.25665e-005,0.00377054)
(-0.00955619,9.57603e-005)
(-0.00205428,-0.00225106)
(0.00379671,-0.00271787)
(-0.00125729,-0.000633952)
(0.000796828,-0.00548449)
(0.00170067,-0.000644856)
(0.0053079,-0.00337067)
(0.0025166,0.00455668)
(0.00440858,-0.00296251)
(0.00332861,0.00561519)
(0.00713061,0.00348795)
(-0.00486489,0.00978036)
(-0.00430955,0.00109993)
(-0.00442742,0.00504222)
(-0.00737744,0.0014526)
(-0.0049828,-0.00489712)
(-0.000394463,-0.00685849)
(-0.000890438,-0.00618168)
(0.00356802,-0.00587584)
(0.00641746,-0.00641548)
(0.0118147,-0.00248627)
(0.0082154,0.00611166)
(0.000436609,0.00630214)
(0.00110045,0.00727227)
(-0.00352936,0.0057802)
(-0.00811984,0.00208274)
(-0.00904693,-0.00398311)
(-0.00639289,-0.00372646)
(-0.0041573,-0.00420071)
(0.00105608,-0.0029152)
(0.00795045,-0.00212169)
(0.00457678,0.00312766)
(0.00217965,0.00611386)
(-0.00102481,-0.00148281)
(0.000513736,-0.000285617)

B)

```
#include <stdlib.h>
#include <cstdio>
#include <cstring>
#include <vector>
#include <math.h>
#include <cmath>
#include <iostream>
#include <fstream>
#include <complex>
```

```
using namespace std;
#define PI 3.14159265358

void split_signal(complex<double> *sig, int len){
	complex<double>* even_sig = new complex<double>[len/2];
	complex<double>* odd_sig = new complex<double>[len/2];

	int even_i = 0, odd_i = 0;
	for ( int i = 0; i < len; i++)
	{
		if ( ( i % 2) == 0){
			even_sig[even_i] = sig[i];
			even_i++;
		}
		else
		{
			odd_sig[odd_i] = sig[i];
			odd_i++;
		}
	}
	int size = len/2;
	memcpy(sig, even_sig, sizeof(complex<double>)*size);
	memcpy(sig+size, odd_sig, sizeof(complex<double>)*size);

	delete[] even_sig;
	delete[] odd_sig;
}
void fast_fourier_transform(complex<double> *sig, double N){
	if (N == 1)
	return;

	int split = N/2;
	complex<double> imag(0,1);
	complex<double> h_n = exp((2*PI*imag)/N);
	complex<double> h = 1;

	split_signal(sig,split*2);

	fast_fourier_transform(sig,split);
	fast_fourier_transform(sig+split,split);

	complex<double> even(0,0), odd(0,0);
	for (int i = 0; i < split;i++){
		even = sig[i];
		odd = sig[i+split];
		sig[i] = even + h*odd;
		sig[i+split] = even - h*odd;
```

```
                h = h*h_n;
        }
}
int main(int argc, char * argv[])
{
        int samples = 1024;
        ifstream infile;
        complex<double>* signal = new complex<double>[samples];
        complex<double>* signal2 = new complex<double>[samples];
        complex<double>* signal3 = new complex<double>[samples];
        complex<double>* signal4 = new complex<double>[samples];
        infile.open("office.ascii");
        for (int i = 0; i < samples; i++)
                infile >> signal[i];
        for (int i = 0; i < samples; i++)
                infile >> signal2[i];
        for (int i = 0; i < samples; i++)
                infile >> signal3[i];
        for (int i = 0; i < samples; i++)
                infile >> signal4[i];
        infile.close();
        fast_fourier_transform(signal,samples);
        fast_fourier_transform(signal2,samples);
        fast_fourier_transform(signal3,samples);
        fast_fourier_transform(signal4,samples);
        ofstream outfile;
        for (int i = 0; i < samples;i++){
                signal2[i] = signal2[i] + signal[i];
                signal3[i] = signal3[i] + signal2[i];
                signal4[i] = signal4[i] + signal3[i];
        }
        outfile.open("result.txt");
        outfile << "Fast Fourier Transform Results:" << endl;
        outfile << "Frame 1 result: " << endl;
        for(int i = 0; i < samples;i++)
        outfile << signal[i] << endl;

        outfile << "Frame 2 result: " << endl;
        for(int i = 0; i < samples;i++)
        outfile << signal2[i] << endl;

        outfile << "Frame 3 result: " << endl;
        for(int i = 0; i < samples;i++)
        outfile << signal3[i] << endl;

        outfile << "Frame 4 result: " << endl;
        for(int i = 0; i < samples;i++)
```

```
        outfile << signal4[i] << endl;

        outfile.close();

        delete[] signal;
};
```