**Program 1**

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <time.h>

/\*Vivian Lam, CS2211 Assignment4: Program 1\*/

/\* Name: Program1

Purpose: find an approximation of the constant pi\*/

int main(void){

//declare variables

double x, y, ratio, mean, standdev, sum, squaresum;

long N =-1;

float num\_in;

//prompts user to input value for N

while (N <0){

printf("Please enter a positive integer N:");

scanf("%d", &N);

}

/\*this is to ensure that the random numbers generated are actually

random and different each time\*/

srand((unsigned) time(NULL));

//loops 10 times

for(int j=0; j<10;j++){

//resets the counter

num\_in = 0;

//loops for N amount of times

for(long i =0; i<N; i++){

/\*generate random numbers between 0.00 and 1.00 to

x and y\*/

x = ((double)rand())/RAND\_MAX;

y = ((double)rand())/RAND\_MAX;

/\*check if this (x,y) coordinate lies inside

shaded quadrant\*/

if((x\*x + y\*y) <= 1){

/\*if the coordinate lies inside the shaded

quadrant then increment the counter that tracks the total number of

coordinates in the quadrant\*/

num\_in++;

}

}

/\*calculate the ratio of points located inside the circle

to the total number of generated points\*/

//multiply the ratio by 4 to find an approximation of pi

ratio = (num\_in/N)\*4;

//prints the value of ratio

printf("The number %d value is: %f \n", j+1, ratio);

//computes the values to find the meand and stnd deviation

sum += ratio;

squaresum += (ratio\*ratio);

}//end outer loop

mean = sum/10; //computes the mean

/\*computes the standard deviation. if the value is

negative then multiply it by -1 to make it positive\*/

standdev = (squaresum/10) - (mean\*mean);

if(standdev<0){

standdev = sqrt(-1\*standdev);

}

else{

standdev = sqrt(standdev);

}

//prints ht mean and standard deviation

printf("Mean: %f \n", mean);

printf("Standard Deviation: %f \n \n", standdev);

//return the exit status 0

return 0;

}

**Test cases**

**obelix.gaul.csd.uwo.ca[34]% prog1**

**Please enter a positive integer N:-123**

**Please enter a positive integer N:10**

**The number 1 value is: 3.600000**

**The number 2 value is: 3.200000**

**The number 3 value is: 2.400000**

**The number 4 value is: 3.200000**

**The number 5 value is: 3.200000**

**The number 6 value is: 3.600000**

**The number 7 value is: 3.600000**

**The number 8 value is: 3.600000**

**The number 9 value is: 3.200000**

**The number 10 value is: 2.800000**

**Mean: 3.240000**

**Standard Deviation: 0.377359**

**obelix.gaul.csd.uwo.ca[35]% prog1**

**Please enter a positive integer N:100**

**The number 1 value is: 3.160000**

**The number 2 value is: 3.320000**

**The number 3 value is: 3.480000**

**The number 4 value is: 3.240000**

**The number 5 value is: 3.160000**

**The number 6 value is: 3.360000**

**The number 7 value is: 3.280000**

**The number 8 value is: 2.880000**

**The number 9 value is: 3.320000**

**The number 10 value is: 3.360000**

**Mean: 3.256000**

**Standard Deviation: 0.155126**

**obelix.gaul.csd.uwo.ca[36]% prog1**

**Please enter a positive integer N:1000**

**The number 1 value is: 3.204000**

**The number 2 value is: 3.060000**

**The number 3 value is: 3.160000**

**The number 4 value is: 3.132000**

**The number 5 value is: 3.172000**

**The number 6 value is: 3.156000**

**The number 7 value is: 3.128000**

**The number 8 value is: 3.216000**

**The number 9 value is: 3.108000**

**The number 10 value is: 3.128000**

**Mean: 3.146400**

**Standard Deviation: 0.043495**

**obelix.gaul.csd.uwo.ca[37]% prog1**

**Please enter a positive integer N:100000**

**The number 1 value is: 3.140080**

**The number 2 value is: 3.143640**

**The number 3 value is: 3.147200**

**The number 4 value is: 3.143240**

**The number 5 value is: 3.147080**

**The number 6 value is: 3.144400**

**The number 7 value is: 3.142760**

**The number 8 value is: 3.144280**

**The number 9 value is: 3.144680**

**The number 10 value is: 3.140600**

**Mean: 3.143796**

**Standard Deviation: 0.002218**

**Please enter a positive integer N:1000000**

**The number 1 value is: 3.141024**

**The number 2 value is: 3.141256**

**The number 3 value is: 3.141648**

**The number 4 value is: 3.141172**

**The number 5 value is: 3.141356**

**The number 6 value is: 3.142480**

**The number 7 value is: 3.139232**

**The number 8 value is: 3.142448**

**The number 9 value is: 3.141092**

**The number 10 value is: 3.142896**

**Mean: 3.141460**

**Standard Deviation: 0.000977**

**obelix.gaul.csd.uwo.ca[39]% prog1**

**Please enter a positive integer N:10000000**

**The number 1 value is: 3.141277**

**The number 2 value is: 3.142175**

**The number 3 value is: 3.141959**

**The number 4 value is: 3.141224**

**The number 5 value is: 3.140803**

**The number 6 value is: 3.141791**

**The number 7 value is: 3.141734**

**The number 8 value is: 3.141261**

**The number 9 value is: 3.141331**

**The number 10 value is: 3.141474**

**Mean: 3.141503**

**Standard Deviation: 0.000388**

Initialize variables:

x; y; ratio; mean; standdev; sum; squaresum; N =-1; num\_in;

Start

NO

YES

NO

YES

YES

NO

YES

NO

Is (x2 + y2) <= 1?

Generate two random numbers between 0.00 and 1.00 and store them in x and y respectively

Is i<N?

Done

i=i+1

Reset num\_in=0;

For loop

j=j+1

Is j<10?

For loop

num\_in = num\_in+1

Done

Done

Store value of input in N

Prompts user for input

Is N<0?

While loop

Compute the mean: sum/10

YES

NO

Square root it

Make it positive

Print the mean and standard deviation

Return 0 (exit status)

Terminate

Is the value under the square root negative?

Compute the value under the square root for standard deviation

**Program 2**

#include <stdio.h>

/\*Vivian Lam, CS2211 Assignment4: Program 2\*/

/\* Name: Program2

Purpose: prints a nxn magic square (a square arrangement of the numbers

1-n\*n in which the sum of the lements in any row, column or diagonal is the

same)\*/

int main(void){

//declare variables

int n = -1;

int row=0, column=0, oldcol=0, oldrow=0;

//creates and inializes the array of integers

int msquare[13][13]={0};

/\*prompts user to input value for n. Loops until number entered is valid\*/

while ((n <1) || (n>13) || (n%2 == 0)){

printf("Please enter an odd positive integer between 1 and 13 (size of magic square):");

scanf("%d", &n);

}

/\*to create the magic square start by placing 1 in the middle of row 0\*/

row =0;

column = n/2;

\*((\*(msquare+row))+column)=1;

/\*get the next position of the pointer (right one and up one). to ensure

that it is in bounds, evaluate the value of the counters\*/

for(int i=2; i<= (n\*n);i++){

/\*variables to store the previous position of the

pointers. used to help place the next element under the

previous one if (if th e next position to place in is occupied\*/

oldcol=column;

oldrow=row;

//increment/decrement counters

column++;

row--;

/\*checking if the counters will go out of array bounds\*/

//MOVING RIGHT

if(column>=n){//counter goes outa bounds

column=0;//reset counter (first column)

}

//MOVING UP

if(row<0){//counter goes outa bounds

row=(n-1);//reset counter (bottom row)

}

/\*if the next position is occupied store number directly

below the previously soted number. otherwise store

normally\*/

/\*not occupied and can insert:\*/

if( (\*((\*(msquare+row))+column)) == 0){ //can insert

\*((\*(msquare+row))+column)= i; //store

}

else{//occupied, insert below previous

/\*set counters equal to previous and make pointer

point here. the +1 in row makes the pointer go a row below\*/

row=oldrow+1;

column=oldcol;

/\*checking if counters go outa bounds\*/

//MOVING RIGHT

if(column>=n){//counter goes outa bounds

column=0;//reset counter

}

//MOVING UP

if(row<0){

row=(n-1);//reset counter

}

\*((\*(msquare+row))+column)= i; /\*storing a row below\*/

}

}//end for

//print the magic square

for(int a=0; a<n; a++){//go to next row

for(int b=0; b<n; b++){//print the row (go to next column)

printf("%6d ", \*((\*(msquare+a))+b) );

}

printf("\n\n");

}

//return the exit status 0

return 0;

}

**Test cases**

**obelix.gaul.csd.uwo.ca[12]% prog2**

**Please enter an odd positive integer between 1 and 13 (size of magic square):1**

**1**

**obelix.gaul.csd.uwo.ca[13]% prog2**

**Please enter an odd positive integer between 1 and 13 (size of magic square):3**

**8 1 6**

**3 5 7**

**4 9 2**

**obelix.gaul.csd.uwo.ca[14]% prog2**

**Please enter an odd positive integer between 1 and 13 (size of magic square):4**

**Please enter an odd positive integer between 1 and 13 (size of magic square):-456**

**Please enter an odd positive integer between 1 and 13 (size of magic square):5**

**17 24 1 8 15**

**23 5 7 14 16**

**4 6 13 20 22**

**10 12 19 21 3**

**11 18 25 2 9**

**obelix.gaul.csd.uwo.ca[15]% prog2**

**Please enter an odd positive integer between 1 and 13 (size of magic square):11**

**68 81 94 107 120 1 14 27 40 53 66**

**80 93 106 119 11 13 26 39 52 65 67**

**92 105 118 10 12 25 38 51 64 77 79**

**104 117 9 22 24 37 50 63 76 78 91**

**116 8 21 23 36 49 62 75 88 90 103**

**7 20 33 35 48 61 74 87 89 102 115**

**19 32 34 47 60 73 86 99 101 114 6**

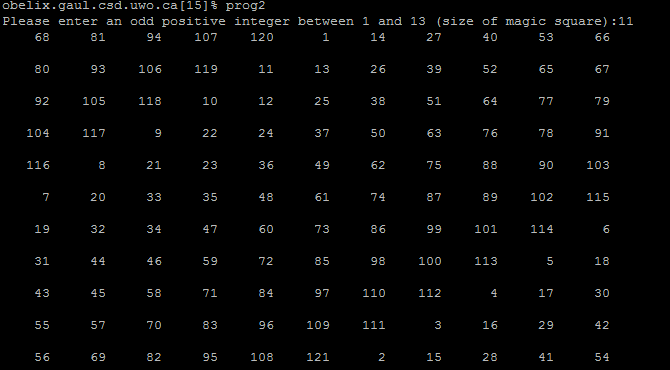
**31 44 46 59 72 85 98 100 113 5 18**

**43 45 58 71 84 97 110 112 4 17 30**

**55 57 70 83 96 109 111 3 16 29 42**

**56 69 82 95 108 121 2 15 28 41 54**

Note that the above looks like the following image, and that the text wrapping of the document borders makes it on a new line.



**obelix.gaul.csd.uwo.ca[16]% prog2**

**Please enter an odd positive integer between 1 and 13 (size of magic square):13**

**93 108 123 138 153 168 1 16 31 46 61 76 91**

**107 122 137 152 167 13 15 30 45 60 75 90 92**

**121 136 151 166 12 14 29 44 59 74 89 104 106**

**135 150 165 11 26 28 43 58 73 88 103 105 120**

**149 164 10 25 27 42 57 72 87 102 117 119 134**

**163 9 24 39 41 56 71 86 101 116 118 133 148**

**8 23 38 40 55 70 85 100 115 130 132 147 162**

**22 37 52 54 69 84 99 114 129 131 146 161 7**

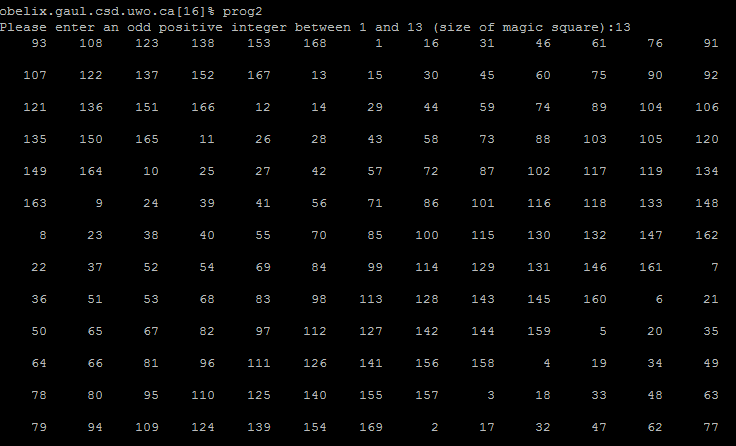
**36 51 53 68 83 98 113 128 143 145 160 6 21**

**50 65 67 82 97 112 127 142 144 159 5 20 35**

**64 66 81 96 111 126 141 156 158 4 19 34 49**

**78 80 95 110 125 140 155 157 3 18 33 48 63**

**79 94 109 124 139 154 169 2 17 32 47 62 77**

Note that the above looks like the following image, and that the text wrapping of the document borders makes it on a new line.

NO

YES

NO

YES

YES

NO

YES

NO

YES

NO

i=i+1

Initialize variables:

n = -1; row=0; column=0; oldcol=0; oldrow=0; i=2; a=0; b=0;

While loop

Start

Done

Is column>=n?

Is row<0?

Is the position occupied?

Store current value of i into that position

Store current value of i below the previous place:

Row = oldrow+1

Col=oldcol

Done

Place 1 in the middle of the first row

For loop

Reset: row=n-1

Reset: column=0

Column = column+1

Row = row-1

Oldcol = column

Oldrow=row

Is i<n2?

Prompts user for input

Store input into variable n

Is n<1 and n>13 and n an even number?

YES

YES

NO

NO

YES

YES

NO

NO

a=a+1

Print element at position [a][b]

b=b+1

Done

Done

Is b<n?

For loop

b=0

Is a<n?

For loop

Reset: row=n-1

Is row<0?

Reset: column=0

Is column>=n?

Return 0 (exit status)

Terminate

**Program 3**

#include <stdio.h>

/\*Vivian Lam, CS2211 Assignment4: Program 3\*/

/\* Name: Program3

Purpose: determines the smallest number of $20, $10, $5, $2, and $1

bills/coins necessary to pay a dollar amount.\*/

/\*prototype for the function\*/

void pay\_amount (int dollars, int \*twenties, int \*tens, int \*fives, int

\*toonies, int \*loonie);

int main(void){

//declares and initialize variables

int dollar\_amount =-1;

int a=0 , b=0, c=0, d=0, e=0;

int \*twenties=&a, \*tens=&b, \*fives=&c, \*toonies=&d, \*loonie=&e;

/\*prompts user to enter an integer value (dollar amount). loops

until the value entered is positive\*/

while (dollar\_amount <0){

printf("Please enter a positive integer for the dollar amount:");

scanf("%d", &dollar\_amount);

}

/\*calls pay\_amount method to modify the values for each of the

pointers\*/

pay\_amount(dollar\_amount, twenties, tens, fives, toonies, loonie);

//prints the results

printf("Number of...\n Twenties: %d \n Tens: %d \n Fives: %d \n Toonies:$

//return the exit status 0

return 0;

}//end main

//pay\_amount function, returns the number of bills of each

void pay\_amount(int dollars, int \*twenties, int \*tens, int \*fives, int

\*toonies, int \*loonie){

/\*modify the values of the variables by finding how many of each

are needed (divide and mod)\*/

\*twenties=dollars /20;

\*tens=(dollars%20)/10;

\*fives=(dollars%10)/5;

\*toonies=(dollars%5)/2;

\*loonie=(dollars%5)%2;

}

**Test cases**

**obelix.gaul.csd.uwo.ca[13]% prog3**

**Please enter a positive integer for the dollar amount:-3456**

**Please enter a positive integer for the dollar amount:0.1876**

**Number of...**

**Twenties: 0**

**Tens: 0**

**Fives: 0**

**Toonies: 0**

**Loonies: 0**

**obelix.gaul.csd.uwo.ca[14]% prog3**

**Please enter a positive integer for the dollar amount:123**

**Number of...**

**Twenties: 6**

**Tens: 0**

**Fives: 0**

**Toonies: 1**

**Loonies: 1**

**obelix.gaul.csd.uwo.ca[15]% prog3**

**Please enter a positive integer for the dollar amount:126**

**Number of...**

**Twenties: 6**

**Tens: 0**

**Fives: 1**

**Toonies: 0**

**Loonies: 1**

**obelix.gaul.csd.uwo.ca[16]% prog3**

**Please enter a positive integer for the dollar amount:138**

**Number of...**

**Twenties: 6**

**Tens: 1**

**Fives: 1**

**Toonies: 1**

**Loonies: 1**

**obelix.gaul.csd.uwo.ca[17]% prog3**

**Please enter a positive integer for the dollar amount:-23**

**Please enter a positive integer for the dollar amount:20**

**Number of...**

**Twenties: 1**

**Tens: 0**

**Fives: 0**

**Toonies: 0**

**Loonies: 0**

**obelix.gaul.csd.uwo.ca[18]% prog3**

**Please enter a positive integer for the dollar amount:59**

**Number of...**

**Twenties: 2**

**Tens: 1**

**Fives: 1**

**Toonies: 2**

**Loonies: 0**

**obelix.gaul.csd.uwo.ca[19]% prog3**

**Please enter a positive integer for the dollar amount:6969**

**Number of...**

**Twenties: 348**

**Tens: 0**

**Fives: 1**

**Toonies: 2**

**Loonies: 0**

**obelix.gaul.csd.uwo.ca[20]% prog3**

**Please enter a positive integer for the dollar amount:8001**

**Number of...**

**Twenties: 400**

**Tens: 0**

**Fives: 0**

**Toonies: 0**

**Loonies: 1**

**obelix.gaul.csd.uwo.ca[21]% prog3**

**Please enter a positive integer for the dollar amount:3**

**Number of...**

**Twenties: 0**

**Tens: 0**

**Fives: 0**

**Toonies: 1**

**Loonies: 1**

Start

YES

NO

Store input into dollar\_amount

Prompts user for input

Modify the values for the variables:

twenties=dollars /20;

tens=(dollars **mod** 20)/10;

fives=(dollars **mod** 10)/5;

toonies=(dollars **mod** 5)/2;

loonie=(dollars **mod** 5)**mod**2;

End function

pay\_amount

Print the values for each of the variables (twenties, tens, fives, toonies, loonie)

pay\_amount

Done

Is dollar\_amount<0?

While loop

Initialize variables:

dollar\_amount=-1; twenties=0; tens=0; fives=0; toonies=0; loonie=0;

Main

Return 0 (exit status)

Terminate