Problem 1

Graphical user interface

Description automatically generated

Problem 2

Graphical user interface

Description automatically generated

Problem 3Graphical user interface

Description automatically generated

All in one

atomGraphical user interface, text

Description automatically generated

Main.c

All in one exe generated

Text

Description automatically generated

Exe output all others

Text

Description automatically generated

#include <stdio.h>

#include "function.h"

int main()

{

p1\_schedule\_input\_intake();

row\_inquiry();

binary\_convert\_intake();

return (0);

}

/\*

@author Bruce Liu

assignment 2

Schedule

run enviorment

ide:

// because it was required

Visual studio 2019 community edition

note:

not bad after getting use to it

I slightly like more atom and conda powershell at the moment.

system info:

Host Name: with held

OS Name: Microsoft Windows 10 Pro

OS Version: 10.0.19042 N/A Build 19042

OS Manufacturer: Microsoft Corporation

OS Configuration: Standalone Workstation

OS Build Type: Multiprocessor Free

Registered Owner: withheld

Registered Organization:

Product ID: withheld

Original Install Date: 12/15/2020, 1:57:08 PM

System Boot Time: 3/7/2021, 4:39:50 PM

System Manufacturer: Micro-Star International Co., Ltd.

System Model: MS-7B79

System Type: x64-based PC

Processor(s): 1 Processor(s) Installed.

[01]: AMD64 Family 23 Model 8 Stepping 2 AuthenticAMD ~3700 Mhz

BIOS Version: American Megatrends Inc. A.30, 4/23/2018

Windows Directory: C:\Windows

System Directory: C:\Windows\system32

Boot Device: \Device\HarddiskVolume1

System Locale: en-us;English (United States)

Input Locale: en-us;English (United States)

Time Zone: (UTC-07:00) Mountain Time (US & Canada)

Total Physical Memory: 32,716 MB

Available Physical Memory: 23,548 MB

Virtual Memory: Max Size: 37,580 MB

Virtual Memory: Available: 25,242 MB

Virtual Memory: In Use: 12,338 MB

Page File Location(s): withheld

Domain: withheld

Logon Server: withheld

Hotfix(s): 10 Hotfix(s) Installed.

[01]: KB4601554

[02]: KB4562830

[03]: KB4570334

[04]: KB4577586

[05]: KB4580325

[06]: KB4586864

[07]: KB4593175

[08]: KB4598481

[09]: KB4601382

[10]: KB5000911

Network Card(s): withheld

DHCP Enabled: No

IP address(es)

withheld

Hyper-V Requirements: withheld

\*/

Functions.h

#ifndef function

#define function

// debug functions

void hello\_world();

void if\_else\_test\_case();

//problem 1 functions

void p1\_schedule\_input\_intake();

void school\_schedule(int day\_number, int hour);

// problem 2 functions

void star\_print\_count();

void space\_print\_count();

void p2\_pyramid();

void row\_inquiry();

//problem 3 functions

void binary\_convert\_intake();

void binary\_convert(int integer\_input);

#endif

Functions.c

#include "function.h"

#include <stdio.h>

void hello\_world()

{

// enviroment testing

printf("Hello World\n\n\n");

}

void if\_else\_test\_case()

{

// had an error

//end result forgot it was military time

int test\_input = 13;

if (test\_input >= 12 && test\_input < 14)

{

printf("proc\n\n\n");

}

else

{

printf("not working\n\n\n");

}

}

P1\_schedule.c

#include "function.h"

#include <stdio.h>

void p1\_schedule\_input\_intake()

{

/\*intialzation declarations

trying a new style and it worked since the compiler doesn't care about white space.

Also it work as demoed here

\*/

int day\_number = 0,

hour = 0 ;

// weekday number input request

printf ("Enter the weekday integer in the range 1-7 (sunday - saturday):\n");

/\* weekday input storing

experimented declaration %i also works

since decimal to interger was slightly confusing

%i is more rational than %d

although they do the same thing

\*/

scanf\_s("%i", &day\_number);

// hour of day input requestmiltary time

printf("Enter the hour of the day interger in the range of 0-23 hours:\n");

//hour of day storage

scanf\_s("%i", &hour);

/\* function abstraction

although it was not needed

I felt that the rest of this function

could be reused so I abstracted the rest

takes in day\_number and hour

\*/

school\_schedule(day\_number, hour);

}

void school\_schedule(int day\_number, int hour)

{

switch (day\_number)

{

case 7:

case 6:

{

//weekend

printf("It is weekend\n"

"I have no classes today.\n"

"Although I may have some students seeking my assistance.\n"

);

break;

}

case 1:

case 3:

{

//monday wednesday output

// weekday output rather than copying and pasteing

if (day\_number == 1)

{

printf("It is a Monday\n");

}

if (day\_number == 3)

{

printf("It is a Wednesday\n");

}

// hour output select

if (hour >= 12 && hour < 14)

{

printf("Currently I have economics of race and gender\n\n\n");

}

else if (hour >= 15 && hour < 17)

{

printf("Currently I have c programing for engineers\n\n\n");

}

else if (hour >= 18 && hour < 20)

{

printf("Currently I have computer science 2\n\n\n");

}

else if (hour >= 20 && hour <= 21 && day\_number == 3)

{

printf("I currently am being a learning assistant for senior design 2\n\n\n");

}

else

{

printf("This time slot is free\n\n\n");

}

break;

}

case 2:

case 4:

{

// output if the input is tuesday or thursday

// weekday selection output rather than copying and pasteing the other stuff

if (day\_number == 2)

{

printf("It is Tuesday.\n\n");

}

if (day\_number == 4)

{

printf("It is Thursday.\n\n");

}

// time slot output select

if (hour >= 12 && hour < 2)

{

printf("I currently have Digital Systems 1.\n\n\n");

}

else

{

printf("This time slot is free. \n"

"Although there is a chance\n"

"I might be assisting someone\n");

}

break;

}

default:

// outputs if a unpredicted action is inputed

printf("Invalid input\n\n");

}

}

P2\_pyramid.c

#include "function.h"

#include <stdio.h>

/\* hypothesis space count negative count for loop

then star count negative for loop

then space count negative for loop

implementing as functions

since I see a chance for code reuse

realoaction of space count by 1 and star by 2

I am going to have a

redunant line of code

for last print row I think.

\*/

void row\_inquiry()

{

// function requests the row count of the pyramid also passes it to P2\_pyramid

int row\_count;

printf("Input the number of row(s) for the pyramid:");

scanf\_s("%i", &row\_count);

p2\_pyramid(row\_count);

}

void p2\_pyramid(int row\_print\_count)

{

// prints the pyramid basied on the requested row\_print\_count requested number

int star\_count = 1;

for(int count = row\_print\_count;

count > 0;

count--

)

{

space\_print\_count(count);

star\_print\_count(star\_count);

space\_print\_count(count);

star\_count += 2;

printf("\n");

}

}

void star\_print\_count(int print\_star\_count)

{

// prints the number of stars requested

for (int count = print\_star\_count;

count > 0;

count--

)

printf("\*");

}

void space\_print\_count(int print\_space\_count)

{

//prints the number of spaces requested

for (int count = print\_space\_count;

count > 0;

count--

)

printf(" ");

}

P3\_binary\_convert.c

#include <stdio.h>

#include "function.h"

// Sorry I must change the apporch

void binary\_convert\_intake()

{

int user\_input = 0;

printf("Input a value to be converted to binary.\n\n");

scanf\_s("%i", &user\_input);

binary\_convert(user\_input);

}

void binary\_convert(int integer\_input)

{

int binary\_memory[64],

count = 0;

// binary conersion and storage

while (integer\_input > 0)

{

if (integer\_input % 2 == 1)

{

binary\_memory[count] = 1;

}

else

{

binary\_memory[count] = 0;

}

count++;

integer\_input /= 2;

}

// inverts the array since the mod tactic needs to be inverted

// this is also why I rebeled against the template

for (int dicount = count - 1; dicount >= 0; dicount--)

{

printf("%i", binary\_memory[dicount]);

}

}

Initial thoughts

You may notice my coding style changed slightly. I found a way to exploit the fact that the compiler ignores white space. So, I only have to call a data type variable once.

Also, I started using %i for integers. Since I for integer makes more sense.

I had an environment brake down after problem 2. So, some of the debug code is not there. I assume you did this as a anti cheat measure. To force use to use Visual studio.

I will have to use my conventional method of anaconda PowerShell and Atom to generate the .exe files since I do not know where they are held in visual studio. I might as well learn to generate a named .exe.

The all in one was the first one I generated. So, it needed a bit of debugging. Namely scanf\_s. It appears that the anaconda PowerShell version needed the \_s removed. I will put them back after I am done generating the .exe files.

Problem 1

Scanf\_s threw me for a loop. Since almost no useful documentation existed. Then I looked at the example files.

I started with switch and case for the weekdays. I remembered that one could have multiple cases assigned to an output to save me some typing. I did this because a college student’s schedule is largely divided into 2 days per class.

I added some if statements for the weekday.

I used if, elseif, else for the hours. I had a issue with this for a moment since I wasn’t thinking in military time initially I had to make a isolated function in function to diagnose this.

I made a function for user intake. Largely Because I did not want to run scanf\_s many times.

Problem 2

This one I went over many iterations in my head. The observation that prompted this is the space count vs the star count.

I needed something so I just started with a single for loop in a function printing a star. The second function was a space.

Afterwards I made another function that used those functions with a 2 to 1 ratio. In a for loop. It worked.

Just like in problem 1 I make a function for user input that took another nested function.

Problem 3

I rebelled against the template. Since I had kind of did this problem before it is on my GitHub orandxor. I also had to use an array.

The error will output as the numbers will be outputted in reverse. I made a new version for this assignment. That is more compressed than the one I had.

A while loop does the conventional mod and divide. With a if and else I pushed a 1 or a zero to a binary memory array.

An inverse for loop combined with an array outputs the solution.