UE20CS152 Week 4

Anirudh Rowjee `PES2UG20CS050`

General File Structure

```
<root directory>
├── main.c
├── Makefile
├── utils.c
└── utils.h
```

Makefile

The Makefile used here is common to all the folders.

```
# Makefile for a simple two-file C program
# Assumes you're running on a UNIX system with gcc
# Author: Anirudh Rowjee PES2UG20CS050

# usage :
# $ make

# To use this Makefile with no Modifications, your
# program must be structured exactly as follows
# (including the filenames):
#
# └── <current directory>
#     ├── main.c
#     ├── Makefile
#     ├── utils.c
#     └── utils.h

CFLAGS=-I.
CC=gcc

# define a header dependencies so that your program re-compiles when
# the header files change
DEPS = utils.h

# define program dependencies so that your program re-compiles when
# the main files change
OBJ = main.o utils.o

# define a rule for the object files, to be cleaned up later
%.o : %.c $(DEPS)
    $(CC) -c -o $@ $< $(CFLAGS)
```

```
# rule for the main files, which cleans up all dependencies
main: $(OBJ)
    $(CC) -o $@ $^ $(CFLAGS)
    $(MAKE) clean

# prevent make from thinking 'clean' is a file
.PHONY: clean

# sanitization
# if you're on Windows replace the command below with
# del *.o
clean:
    rm *.o
```

Task 1

Write a function to display the elements of an array in the reverse order by using multiple files, and through

1. Index
2. Pointer

Code

main.c

```c
#include <stdio.h>
#include "utils.h"

/*
 * Program to traverse an array in reverse order using both pointers
 * and indices.
 */

int main()
{
    // get the array from the user
    int element_count;
    printf("Enter the Number of elements you want in your array > ");
    scanf("%d", &element_count);

    int user_array[element_count];
    for(int i = 0; i < element_count; i++)
    {
        printf("Enter element number %d > ", i+1);
        scanf("%d", &user_array[i]);
    }

    printf("Traversing by Index > \n");
    ReverseTraverseByIndex(user_array, element_count);
```

```
    printf("Traversing by Index > \n");
    ReverseTraverseByPointer(user_array, element_count);
    return 0;
}
```

utils.c

```
#include "utils.h"
#include <stdio.h>

void ReverseTraverseByIndex(int* myArray, int element_count)
{
    // function to traverse an array by index
    for (int i = 0; i < element_count; i++)
    {
        printf("%d ", myArray[element_count - 1 - i]);
    }
    printf("\n");
}

void ReverseTraverseByPointer(int* myArray, int element_count)
{
    // find the address of the final element of the array
    int* final_address = myArray + element_count - 1;
    // attempt linear traversal to make sure pointer math is right
    do {
        printf("%d ", *final_address);
        final_address--;
    }
    while (final_address != myArray - 1);
    printf("\n");
}
```

utils.h

```
// header file: utils.h
void ReverseTraverseByIndex(int* myArray, int element_count);
void ReverseTraverseByPointer(int* myArray, int element_count);
```

Console

```
nidavellir :: UE20CS152/week3/t1 » ./main
Enter the Number of elements you want in your array > 3
Enter element number 1 > 1
Enter element number 2 > 2
Enter element number 3 > 3
Traversing by Index >
3 2 1
```

```
Traversing by Index >
3 2 1
```

Task 2

Write a function to find the factorial of a number using recursion and use it to find C(n,r). Use multiple files.

Code

main.c

```c
#include <stdio.h>
#include "utils.h"


int main()
{
    int n, r;
    printf("Enter the numbers N and R separated by a space.\n");
    scanf("%d %d", &n, &r);
    int answer = factorial(n) / (factorial(n-r) * factorial(r));
    printf("The answer nCr is > %d\n", answer);
    return 0;
}
```

utils.c

```c
#include <stdio.h>
#include "utils.h"


int factorial(int base)
{
    if (base <= 1)
    {
        return 1;
    }
    else
    {
        return base * factorial(base - 1);
    }
}
```

utils.h

```c
// header file

int factorial(int base);
```

Console

```
nidavellir :: UE20CS152/week3/t2 » ./main
Enter the numbers N and R separated by a space.
5 3
The answer nCr is > 10
nidavellir :: UE20CS152/week3/t2 » ./main
Enter the numbers N and R separated by a space.
6 3
The answer nCr is > 20
```

Task 3

Write a program to print the unique elements of an array.

Code

main.c

```c
#include <stdio.h>
#include "utils.h"

int main()
{
    int count;
    printf("Enter the number of elements you want to add > ");
    scanf("%d", &count);

    int sample[count];
    int target[count];

    for (int i = 0; i < count; i++)
    {
        printf("Enter element %d > ", i+1);
        scanf("%d", &sample[i]);
    }

    int unique_count = LoadUniqueElements(sample, target, count);

    printf("The Unique Elements of the Array are > \n");
    for (int i = 0; i < unique_count; i++)
    {
        printf("%d ", target[i]);
    }
```

```
    printf("\n");
}
```

utils.c

```
#include <stdio.h>
#include <stdbool.h>
#include "utils.h"

int LoadUniqueElements(int* sample, int* target, int element_count)
{
    // function that takes pointers to two arrays of the same
    // length and loads the unique elements of sample
    // into target
    // returns the number of unique elements
    int size = element_count;
    int unique_top_index = 0;

    for (int i = 0; i < size; i++)
    {
        int sample_element = sample[i];
        bool unique = true;
        for (int j = 0; j < size; j++)
        {
            if (target[j] == sample_element)
            {
                unique = false;
                break;
            }
        }
        // if unique is still true at this point, we make it a new
        // element in the list of unique numbers
        if (unique)
        {
            target[unique_top_index] = sample_element;
            unique_top_index++;
        }
    }
    return unique_top_index;
}
```

utils.h

```
// header file

int LoadUniqueElements(int* sample, int* target, int element_count);
```

Console

```
nidavellir :: UE20CS152/week3/t3 » ./main
Enter the number of elements you want to add > 3
Enter element 1 > 1
Enter element 2 > 2
Enter element 3 > 1
1 2
nidavellir :: UE20CS152/week3/t3 » ./main
Enter the number of elements you want to add > 5
Enter element 1 > 1
Enter element 2 > 2
Enter element 3 > 1
Enter element 4 > 2
Enter element 5 > 1
1 2
nidavellir :: UE20CS152/week3/t3 » ./main
Enter the number of elements you want to add > 5
Enter element 1 > 1
Enter element 2 > 2
Enter element 3 > 1
Enter element 4 > 1
Enter element 5 > 3
1 2 3
```

Task 4

Write a C program to calculate the power of any number using recursion and multiple files.

Code

main.c

```c
#include <stdio.h>
#include "utils.h"


int main()
{
    int x, y;
    printf("Enter the two numbers separated by a space (x,y) > ");
    scanf("%d %d", &x, &y);
    int answer = custom_pow(x, y);
    printf("%d to the power of %d is > %d\n", x, y, answer);
    return 0;
}
```

utils.c

```c
#include <stdio.h>
#include "utils.h"


int custom_pow(int x, int y)
{
    // function to recursively calculate the power of one number to the other
    // this function recursively calculates x to the power of y
    if (y <= 1)
    {
        return x;
    }
    else
    {
        return x * custom_pow(x, y-1);
    }
}
```

`utils.h`

```c
// header file

int custom_pow(int x, int y);
```

Console

```
nidavellir :: UE20CS152/week3/t4 » ./main
Enter the two numbers separated by a space (x,y) > 3 2
3 to the power of 2 is > 9
nidavellir :: UE20CS152/week3/t4 » ./main
Enter the two numbers separated by a space (x,y) > 10 10
10 to the power of 10 is > 1410065408
nidavellir :: UE20CS152/week3/t4 » ./main
Enter the two numbers separated by a space (x,y) > 10 4
10 to the power of 4 is > 10000
```

Task 5

Write a function to check whether a given number is Prime, and use that to find the next prime number greater than a given number.

Code

`main.c`

```c
#include <stdio.h>
#include <stdbool.h>
#include "utils.h"

int main()
{
    // define the maximium size we can check for
    const int MAX_VAL = 10000;
    int test;
    printf("Enter the number to check the primality of > ");
    scanf("%d", &test);

    bool sieve[MAX_VAL];
    // set every element to True
    for (int i = 0; i < MAX_VAL; i++)
    {
        sieve[i] = true;
    }

    initialize_sieve(sieve, MAX_VAL);
    check_number(test, sieve);
    int next_number = get_next_prime(test, sieve, MAX_VAL);
    printf("The next prime number is %d\n", next_number);

    return 0;
}
```

utils.c

```c
#include <stdio.h>
#include <stdbool.h>
#include "utils.h"


void initialize_sieve(bool* sieve, int MAX_VAL)
{
    // initialize the sieve of eratosthenes till the maximum integer limit
    // stored in INT_MAX, and expects the array sieve to have the said
    // number of elements.

    // iterate through the list as every number is a potential divisor
    // we index the outer array from one, making the offset plus one
    for (int i = 2; i < MAX_VAL; i++)
    {
        // at this point of time, we mark off every number here onward that
        // is a multiple of this current number
        for (int j = i+1; j < MAX_VAL; j++)
        {
            if (j % i == 0)
            {
                sieve[j-1] = false;
            }
```

```
        }
    }
}


void check_number(int number, bool* sieve)
{
    // function to check if the number at number-1 is prime or not
    // it returns the index of this number regardless
    printf("The number %d is %s\n", number,
            sieve[number-1] ? "Prime" : "not prime"
            );
}

int get_next_prime(int number, bool* sieve, int MAX_VAL)
{
    for (int i = number; i < MAX_VAL; i++)
    {
        if (sieve[i])
        {
            return i+1;
        }
    }
}
```

utils.h

```
// header file

void initialize_sieve(bool* sieve, int MAX_VAL);
void check_number(int number, bool* sieve);
int get_next_prime(int number, bool* sieve, int MAX_VAL);
```

Console

```
nidavellir :: UE20CS152/week3/t5 » ./main
Enter the number to check the primality of > 1379
The number 1379 is not prime
The next prime number is 1381
nidavellir :: UE20CS152/week3/t5 » ./main
Enter the number to check the primality of > 2041
The number 2041 is not prime
The next prime number is 2053
nidavellir :: UE20CS152/week3/t5 » ./main
Enter the number to check the primality of > 9133
The number 9133 is Prime
The next prime number is 9137
```

Practice Problem 1

Write a program to find the largest and smallest elements of an array.

Code

main.c

```c
#include <stdio.h>
#include "utils.h"


int main()
{
    // write a program to find the smallest and largest elements in an array.
    int limit;
    printf("Enter the number of elements in the array > ");
    scanf("%d", &limit);

    int numbers[limit];

    for (int i = 0; i < limit; i++)
    {
        printf("Enter the number at position %d > ", i+1);
        scanf("%d", &numbers[i]);
    }

    int smallest = FindSmallestElement(numbers, limit);
    int largest = FindLargestElement(numbers, limit);

    printf("The Smallest Element in the list is %d\n", smallest);
    printf("The Largest Element in the list is %d\n", largest);

    return 0;
}
```

utils.c

```c
#include <stdio.h>
#include "utils.h"
#include <limits.h>

int FindSmallestElement(int* numbers, int length)
{
    // function to return the smallest element of an array
    int smallest = INT_MAX;
    int size = length;

    for (int i = 0; i < size; i++)
    {
```

```
        if (numbers[i] < smallest)
        {
            smallest = numbers[i];
        }
    }
    return smallest;
}


int FindLargestElement(int* numbers, int length)
{
    // function to return the smallest element of an array
    int greatest = INT_MIN;
    int size = length;

    for (int i = 0; i < size; i++)
    {
        if (numbers[i] > greatest)
        {
            greatest = numbers[i];
        }
    }
    return greatest;
}
```

utils.h

```
// header file

int FindLargestElement(int* numbers, int length);
int FindSmallestElement(int* numbers, int length);
```

Console

```
nidavellir :: UE20CS152/week3/pp1 » ./main
Enter the number of elements in the array > 5
Enter the number at position 1 > 5
Enter the number at position 2 > 3
Enter the number at position 3 > 3
Enter the number at position 4 > 2
Enter the number at position 5 > 4
The Smallest Element in the list is 2
The Largest Element in the list is 5
nidavellir :: UE20CS152/week3/pp1 » ./main
Enter the number of elements in the array > 5
Enter the number at position 1 > 1
Enter the number at position 2 > 2
Enter the number at position 3 > 1
Enter the number at position 4 > 1
Enter the number at position 5 > 1
```

```
The Smallest Element in the list is 1
The Largest Element in the list is 2
```

Practice Problem 2

Write a program to populate an array with the fibonacci series.

Code

main.c

```c
#include <stdio.h>
#include "utils.h"

int main()
{
    int limit;

    printf("Enter the Count of elements of the Fibonacci Array > ");
    scanf("%d", &limit);

    int fibArray[limit];

    PropagateFibonacci(fibArray, limit);

    for (int i = 0; i < limit; i++)
    {
        printf("%d ", fibArray[i]);
    }
    printf("\n");

    return 0;
}
```

utils.c

```c
#include <stdio.h>
#include "utils.h"

void PropagateFibonacci(int* number_array, int limit)
{
    // function to fill the array number_array with fibonacci numbers
    // till we reach the nth fibonacci number
    if (number_array[0] != 0 || number_array[1] != 1)
    {
        number_array[0] = 0;
        number_array[1] = 1;
    }
```

```
    for (int i = 2; i < limit; i++)
    {
        number_array[i] = number_array[i-1] + number_array[i-2];
    }
}
```

utils.h

```
// header file
void PropagateFibonacci(int* number_array, int limit);
```

Console

```
nidavellir :: UE20CS152/week3/pp2 » ./main
Enter the Count of elements of the Fibonacci Array > 5
0 1 1 2 3
nidavellir :: UE20CS152/week3/pp2 » ./main
Enter the Count of elements of the Fibonacci Array > 10
0 1 1 2 3 5 8 13 21 34
```