0

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

/\*\*

 \* main - Entry point

 \*

 \* Return: Always 0 (success)

 \*/

int main(void)

{

printf("\_putchar\n");

return (0);

}

1

#include "main.h"

#include <unistd.h>

/\*\*

 \* print\_alphabet - Entry point

 \* Return:0

 \*/

void print\_alphabet(void)

{

        char c = 'a';

        while (c <= 'z')

        {

                \_putchar(c);

                c++;

        }

        \_putchar('\n');

}

2

#include "main.h"

/\*\*

 \* print\_alphabet\_x10 - Print lowercase alphabet 10 times

 \*/

void print\_alphabet\_x10(void)

{

        char num, letter;

        for (num = 0; num <= 9; num++)

        {

                for (letter = 'a'; letter <= 'z'; letter++)

                {

                        \_putchar(letter);

                }

                \_putchar('\n');

        }

}

3

#include "main.h"

/\*\*

 \* \_islower - Checks for lowercase alphabet

 \* @c: Character to be checked

 \* Return: if character is lowercase, otherwise

 \*/

int \_islower(int c)

{

        if (c >= 'a' && c <= 'z')

                return (1);

        else

                return (0);

}

Task 4

#include "main.h"

/\*\*

 \* \_isalpha - Checks for alphabetic order

 \*

 \* @c: alpabet to check for

 \*

 \* Return: 1 if c is a letter, 0 otherwise

 \*/

int \_isalpha(int c)

{

        return ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z'));

}

Task 5

#include "main.h"

/\*\*

 \* print\_sign - prints the sign of a numver

 \*

 \* @n: the int to check

 \*

 \* Return: 1 and prints + if n is greater than zero

 \*

 \* 0 and prints 0 if n is zero

 \*

 \* -1 and prints - if n is less than zero

 \*/

int print\_sign(int n)

{

        if (n > 0)

        {

                \_putchar('+');

                return (1);

        }

        else if (n == 0)

        {

                \_putchar(48);

                return (0);

        }

        else if (n < 0)

        {

                \_putchar('-');

        }

        return (-1);

}

0x02-functions\_nested\_loops

Task 0

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

/\*\*

 \* main - Entry point

 \*

 \* Return: Always 0 (success)

 \*/

int main(void)

{

printf("\_putchar\n");

return (0);

}

Task 1

#include "main.h"

#include <unistd.h>

/\*\*

 \* print\_alphabet - Entry point

 \* Return:0

 \*/

void print\_alphabet(void)

{

        char c = 'a';

        while (c <= 'z')

        {

                \_putchar(c);

                c++;

        }

        \_putchar('\n');

}

Task 2

#include "main.h"

/\*\*

 \* print\_alphabet\_x10 - Print lowercase alphabet 10 times

 \*/

void print\_alphabet\_x10(void)

{

        char num, letter;

        for (num = 0; num <= 9; num++)

        {

                for (letter = 'a'; letter <= 'z'; letter++)

                {

                        \_putchar(letter);

                }

                \_putchar('\n');

        }

}

Task 3

#include "main.h"

/\*\*

 \* \_islower - Checks for lowercase alphabet

 \* @c: Character to be checked

 \* Return: if character is lowercase, otherwise

 \*/

int \_islower(int c)

{

        if (c >= 'a' && c <= 'z')

                return (1);

        else

                return (0);

}

Task 4

#include "main.h"

/\*\*

 \* \_isalpha - Checks for alphabetic order

 \*

 \* @c: alpabet to check for

 \*

 \* Return: 1 if c is a letter, 0 otherwise

 \*/

int \_isalpha(int c)

{

        return ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z'));

}

Task 5

#include "main.h"

/\*\*

 \* print\_sign - prints the sign of a numver

 \*

 \* @n: the int to check

 \*

 \* Return: 1 and prints + if n is greater than zero

 \*

 \* 0 and prints 0 if n is zero

 \*

 \* -1 and prints - if n is less than zero

 \*/

int print\_sign(int n)

{

        if (n > 0)

        {

                \_putchar('+');

                return (1);

        }

        else if (n == 0)

        {

                \_putchar(48);

                return (0);

        }

        else if (n < 0)

        {

                \_putchar('-');

        }

        return (-1);

}

Task 6

#include "main.h"

/\*\*

 \* \_abs - computes the absolute value of an integrer

 \*

 \* @n: the int to check

 \*

 \* Return: the absolute value of int

 \*

 \*/

int \_abs(int n)

{

        if (n >= 0)

        {

                return (n);

        }

        return (-n);

}

Task 7

#include "main.h"

/\*\*

 \* print\_last\_digit - prints the last digit of a number

 \*

 \* @n: the int to extract the last digit from

 \*

 \* Return: value of the last digit

 \*

 \*/

int print\_last\_digit(int n)

{

        int a;

        if (n < 0)

                n = -n;

        a = n % 10;

        if (a < 0)

                a = -a;

        \_putchar(a + '0');

        return (a);

}

Task 8

#include "main.h"

/\*\*

 \* jack\_bauer - prints every minute of the day of Jack Bauer

 \*

 \* starting from 00:00 to 23:59

 \*

 \*/

void jack\_bauer(void)

{

        int i, j;

        i = 0;

        while (i < 24)

        {

                j = 0;

                while (j < 60)

                {

                        \_putchar((i / 10) + '0');

                        \_putchar((i % 10) + '0');

                        \_putchar(':');

                        \_putchar((j / 10) + '0');

                        \_putchar((j % 10) + '0');

                        \_putchar('\n');

                        j++;

                }

                i++;

        }

}

Task 9

#include "main.h"

/\*\*

 \* times\_table - check description

 \* Description: It prints 9 times table starting with 0

 \* Return: Nothing.

 \*/

void times\_table(void)

{

        int i, j, n;

        for (i = 0; i <= 9; i++)

        {

                for (j = 0; j <= 9; j++)

                {

                        n = i \* j;

                        if ((n / 10) == 0)

                        {

                                if (j != 0)

                                        \_putchar(' ');

                                \_putchar(n + '0');

                                if (j == 9)

                                        continue;

                                \_putchar(',');

                                \_putchar(' ');

                        }

                        else

                        {

                                \_putchar((n / 10) + '0');

                                \_putchar((n % 10) + '0');

                                if (j == 9)

                                        continue;

                                \_putchar(',');

                                \_putchar(' ');

                        }

                }

                \_putchar('\n');

        }

}

Task 10

#include "main.h"

/\*\*

 \* add - adds two integers and returnsthe result

 \*

 \* @x: number being added

 \*

 \* @y: number being added

 \*

 \* Return: Always 0.

 \*

 \*/

int add(int x, int y)

{

        return (x + y);

}

Task 11

#include "main.h"

#include <stdio.h>

/\*\*

 \* print\_to\_98 - prints all natural numbers from n to 98

 \*

 \* @n: starting point

 \*

 \* Return: returns nothing

 \*

 \*/

void print\_to\_98(int n)

{

        if (n <= 98)

        {

                for (n = n; n <= 97; n++)

                        printf("%d, ", n);

        }

        else

                for (n = n; n > 98; n--)

                        printf("%d, ", n);

        printf("98\n");

}

Task 12

#include "main.h"

/\*\*

 \* print\_times\_table - prints the n times table, starting with 0

 \* @n: number of the times table

 \*/

void print\_times\_table(int n)

{

        int i, j, k;

        if (n >= 0 && n <= 15)

        {

                for (i = 0; i <= n; i++)

                {

                        for (j = 0; j <= n; j++)

                        {

                                k = j \* i;

                                if (j == 0)

                                {

                                        \_putchar(k + '0');

                                } else if (k < 10 && j != 0)

                                {

                                        \_putchar(',');

                                        \_putchar(' ');

                                        \_putchar(' ');

                                        \_putchar(' ');

                                        \_putchar(k + '0');

                                } else if (k >= 10 && k < 100)

                                {

                                        \_putchar(',');

                                        \_putchar(' ');

                                        \_putchar(' ');

                                        \_putchar((k / 10) + '0');

                                        \_putchar((k % 10) + '0');

                                } else if (k >= 100)

                                {

                                        \_putchar(',');

                                        \_putchar(' ');

                                        \_putchar((k / 100) + '0');

                                        \_putchar(((k / 10) % 10) + '0');

                                        \_putchar((k % 10) + '0');

                                }

                        }

                        \_putchar('\n');

                }

        }

}

Task 13

#include <stdio.h>

/\*\*

 \* main - computes and prints the sum of all the multiples

 \* of 3 or 5 below 1024

 \* Return: Always 0 (success)

 \*/

int main(void)

{

        unsigned long int sum3, sum5, sum;

        int i;

        sum3 = 0;

        sum5 = 0;

        sum = 0;

        for (i = 0; i < 1024; ++i)

        {

                if ((i % 3) == 0)

                {

                        sum3 = sum3 + i;

                } else if ((i % 5) == 0)

                {

                        sum5 = sum5 + i;

                }

        }

        sum = sum3 + sum5;

        printf("%lu\n", sum);

        return (0);

}

Task 14

#include <stdio.h>

/\*\*

 \* main - prints the first 50 Fibonacci numbers, starting with 1 and 2

 \* followed by a new line

 \* Return: Always 0 (Success)

 \*/

int main(void)

{

        long int i, j, k, next;

        j = 1;

        k = 2;

        for (i = 1; i <= 50; ++i)

        {

                if (j != 20365011074)

                {

                        printf("%ld, ", j);

                } else

                {

                        printf("%ld\n", j);

                }

                next = j + k;

                j = k;

                k = next;

        }

        return (0);

}

Task 15

#include <stdio.h>

/\*\*

 \* main - finds and prints the sum of the even-valued terms

 \* followed by a new line

 \* Return: Always 0 (Success)

 \*/

int main(void)

{

        int i;

        unsigned long int j, k, next, sum;

        j = 1;

        k = 2;

        sum = 0;

        for (i = 1; i <= 33; ++i)

        {

                if (j < 4000000 && (j % 2) == 0)

                {

                        sum = sum + j;

                }

                next = j + k;

                j = k;

                k = next;

        }

        printf("%lu\n", sum);

        return (0);

}

Task 16

#include <stdio.h>

/\*\*

 \* numLength - returns the lenth of string

 \* @num : operand number

 \* Return: number of digits

 \*/

int numLength(int num)

{

        int length = 0;

        if (!num)

        {

                return (1);

        }

        while (num)

        {

                num = num / 10;

                length += 1;

        }

        return (length);

}

/\*\*

 \* main - prints the first 98 fibonaci sequences

 \* Return: 0

 \*/

int main(void)

{

        unsigned long f1 = 1, f2 = 2, tmp, mx = 100000000, f1o = 0, f2o = 0, tmpo = 0;

        short int i = 1, initial0s;

        while (i <= 98)

        {

                if (f1o > 0)

                        printf("%lu", f1o);

                initial0s = numLength(mx) - 1 - numLength(f1);

                while (f1o > 0 && initial0s > 0)

                {

                        printf("%i", 0);

                        initial0s--;

                }

                printf("%lu", f1);

                tmp = (f1 + f2) % mx;

                tmpo = f1o + f2o + (f1 + f2) / mx;

                f1 = f2;

                f1o = f2o;

                f2 = tmp;

                f2o = tmpo;

                if (i != 98)

                        printf(", ");

                else

                        printf("\n");

                i++;

        }

        return (0);

}