```
----------------------------------
AFF_A-----------------------------------------------

#include <unistd.h>

int main(int ac, char **av)
{
        int i;
        i = 0;
        if(ac == 2)
        {
                while(av[1][i] != '\0')
                {
                        if (av[1][i] == 'a')
                        {
                                write(1, "a", 1);
                                break;
                        }
                        i++;
                }
                write(1, "\n", 1);
        }
        else
                write(1,"a\n", 2);
        return 0;
}

---------------------------
AFF_Z-------------------------------------------------------
#include <unistd.h>

int             main(int ac, char **av)
{
        (void)ac;
        (void)av;
        write(1, "z\n", 2);
        return (0);
}

---------------------------
FİZZBUZZ----------------------------------------------------

void ft_putchar(char c)
{
        write(1, &c, 1);
}
void yazaq(int say)
{
        if (say < 10)
```

```
                ft_putchar(say + '0');
        else
        {
                yazaq(say / 10);
                yazaq(say % 10);
        }
}
void    fizbuz(void)
{
        int say;
        say = 1;
        while (say <= 100)
        {
                if (say % 3 == 0 && say % 5 == 0)
                        write(1, "fizzbuzz", 8);
                else if (say % 3 == 0)
                        write(1, "fizz", 4);
                else if (say % 5 == 0)
                        write(1 , "buzz", 4);
                else
                        yazaq(say);
                say++;
                write(1, "\n", 1);
        }
}
int main(void)
{
        fizbuz();
}
```

ULTSTR----------

```
#include<unistd.h>

int main(int argc, char **argv)
{
        int a;
        char b;
        a = 0;
        if (argc == 2)
        {
                while (argv[1][a])
                {
                        b = argv[1][a];
                        if (argv[1][a] <= 'A' && argv[1][a] >= 'Z')
                                b += 32;
                        if (argv[1][a] <= 'a' && argv[1][a] >= 'z')
                                b -= 32;
                        write(1,&b,1);
```

```
                        a++;
                }
        }
        write(1,"\n", 1);
        return 0;
}
```

REVPRİNT

```
#include <unistd.h>
#include <stdio.h>
int     ft_strlen(char *str)
{
        int     i;
        i = 0;
        while(str[i] != '\0')
                i++;
        return (i);
}
char *ft_rev_print(char *str)
{
        int     i;

        i = ft_strlen(str);
        i--;
        while (i >= 0)
        {
                write(1, &str[i], 1);
                i--;
        }
        return(str);
}
```

LASTPARAM

```
#include <unistd.h>

int             main(int ac, char **av)
{
        int i;

        i = 0;
        if (ac > 1)
        {
//Bunu unutma   ac--;
                while (av[ac][i] != '\0')
                {
                        write(1, &av[ac][i], 1);
                        i++;
```

```
                }
        }
        write(1, "\n", 1);
        return (0);
}
```

STRCPY

```
#include <unistd.h>

char    *ft_strcpy(char *s1, char *s2)
{
        int i;

        i = 0;
        while (s2[i] != '\0')
        {
                s1[i] = s2[i];
                i++;
        }
        s1[i] = '\0';
        return (s1);
}
```

UNION

```
#include <unistd.h>

int             check_doubles2(char *str, char c)
{
        int i;

        i = 0;
        while (str[i] != '\0')
        {
                if (str[i] == c)
                        return (0);
                i++;
        }
        return (1);
}

int             check_doubles1(char *str, char c, int pos)
{
        int i;

        i = 0;
        while (i < pos)
        {
                if (str[i] == c)
```

```
                        return (0);
                i++;
        }
        return (1);
}

void    ft_union(char *str, char *str1)
{
        int     i;

        i = 0;
        while (str[i] != '\0')
        {
                if (check_doubles1(str, str[i], i) == 1)
                        write(1, &str[i], 1);
                i++;
        }
        i = 0;
        while (str1[i] != '\0')
        {
                if (check_doubles2(str, str1[i]) == 1)
                {
                        if (check_doubles1(str1, str1[i], i) == 1)
                                write(1, &str1[i], 1);
                }
                i++;
        }
}

int             main(int ac, char **av)
{
        if (ac == 3)
                ft_union(av[1], av[2]);
        write(1, "\n", 1);
        return (0);
}

_________________________________
FİRSTPARAM-----------------------------------------
#include<unistd.h>

int main(int ac, char **av)
{
        int a=0;
        if (ac > 1)
        {
                while (av[1][a])
                {
                        write(1, &av[1][a], 1);
                        a++;
```

```
                }
                write(1, "\n", 1);
        }
        else
                write(1, "\n", 1);
}
```

PUTSTR

```
void    ft_putstr(char *str)

{
        int             a;

        a = 0;
        while (str[a] != '\0')
        {
                write(1, &str[a], 1);
                a++;
        }
}
```

STRLEN

```
#include<unistd.h>

int     ft_strlen(char *str)
{
        int             p;

        p = 0;
        while (str[p] != '\0')
        {
                p++;
        }
        return (p);
}
```

SWAP

```
void    ft_swap(int *a, int *b)
{
        int aux;

        aux = *a;
        *a = *b;
        *b = aux;
}
```

ROT_13

```
#include <unistd.h>
```

```
void    ft_putchar(char c)
{
        write(1, &c, 1);
}

int             rot_13(char c)
{
        if ((c >= 'A' && c <= 'M') || (c >= 'a' && c <= 'm'))
                c += 13;
        else if ((c >= 'N' && c <= 'Z') || (c >= 'n' && c <= 'z'))
                c -= 13;
        return (c);
}

int             main(int ac, char **av)
{
        if (ac == 2)
                while (*av[1])
                        ft_putchar(rot_13(*av[1]++));
        ft_putchar('\n');
        return (0);
}
```

SEARCH_AND_REPLACE

```
#include <unistd.h>

int main(int argc, char **argv)
{
        int c;

        c = 0;
        if (argc != 4)
        {
                write(1, "\n", 1);
                return (0);
        }

        if (argv[2][1] != '\0' || argv[3][1] != '\0')
        {
                write(1, "\n", 1);
                return 0;
        }

        while (argv[1][c] != '\0')
        {
                if (argv[1][c] == argv[2][0])
                        argv[1][c] = argv[3][0];
                write(1, &argv[1][c], 1);
                c++;
        }
```

```
        write(1, "\n", 1);
        return 0;
}
----------------------------------
ROTANE-------------------------------------------
#include <unistd.h>

int             main(int argc, char *argv[])
{
        int     i = 0;
        char    ltr = argv[1][i];

        if (argc == 2)
        {
                while (argv[1][i])
                {
                        if (argv[1][i] >= 'A' && argv[1][i] <= 'Y')
                                ltr += 1;
                        if (argv[1][i] >= 'a' && argv[1][i] <= 'y')
                                ltr += 1;
                        if (argv[1][i] == 'Z' || argv[1][i] == 'z')
                                ltr -= 25;
                        write(1, &ltr, 1);
                        i += 1;
                }
        }
        write(1, "\n", 1);
        return (0);
}
----------------------------------
STRCMP-------------------------------------------
int             ft_strcmp(char *s1, char *s2)
{
        int i;

        i = 0;
        while (s1[i] != '\0' && s2[i] != '\0' && s1[i] == s2[i])
                i++;
        return (s1[i] - s2[i]);
}
------------------------------
STRREV-------------------------------------------------
int             ft_strlen(char *str)
{
        int i;

        i = 0;
        while (str[i] != '\0')
                i++;
```

```
	return (i);
}

char	*ft_strrev(char *str)
{
	int i;
	int len;
	char tmp;

	i = 0;
	len = ft_strlen(str) - 1; // - 1i UNUTMA SAKIN
	while (len > i)
	{
		tmp = str[i];
		str[i] = str[len];
		str[len] = tmp;
		i++;
		len--;
	}
	return (str);
}
-------------------------
İNTER-------------------------------------------------------------
#include <unistd.h>

int		check_doubles(char *str, char c, int pos)
{
	int i;

	i = 0;
	while (i < pos)
	{
		if (str[i] == c)
			return (0);
		i++;
	}
	return (1);
}

int		main(int ac, char **av)
{
	int i;
	int i2;

	i = 0;
	if (ac == 3)
	{
		while (av[1][i] != '\0')
		{
			i2 = 0;
```

```
                        while (av[2][i2] != '\0')
                        {
                                if (av[1][i] == av[2][i2])
                                {
                                        if (check_doubles(av[1],
av[1][i], i))
                                        {
                                                write(1, &av[1][i],
1);
                                                break ;
                                        }
                                }
                                i2++;
                        }
                        i++;
                }
        }
        write(1, "\n", 1);
        return (0);
}
-----------------------------------
WDMMATCH-----------------------------------------------
#include <unistd.h>

void    ft_putstr(char *str)
{
        int i;

        i = 0;
        while (str[i] != '\0')
        {
                write(1, &str[i], 1);
                i++;
        }
}

int             ft_strlen(char *str)
{
        int i;

        i = 0;
        while (str[i] != '\0')
                i++;
        return (i);
}

int             main(int ac, char **av)
{
        int i;
        int i2;
```

```
	int wdlen;

	i = 0;
	i2 = 0;
	wdlen = 0;
	if (ac == 3)
	{
		while (av[1][i] != '\0')
		{
			while (av[2][i2] != '\0')
			{
				if (av[1][i] == av[2][i2])
				{
					wdlen++;
					break ;
				}
				i2++;
			}
			i++;
		}
		if (wdlen == ft_strlen(av[1]))
			ft_putstr(av[1]);
	}
	write(1, "\n", 1);
	return (0);
}
-----------------------------
ATOİ-----------------------------------------------------

int		ft_atoi(const char *str)
{
	int i;
	int sign;
	int result;

	i = 0;
	sign = 1;
	result = 0;
	while (str[i] == 32 || (str[i] >= 9 && str[i] <= 13))
		i++;
	if (str[i] == '-')
	{
		sign = -1;
		i++;
	}
	else if (str[i] == '+')
		i++;
	while (str[i] != '\0' && str[i] >= '0' && str[i] <= '9')
	{
		result *= 10;
```

```
                result += str[i] - '0';
                i++;
        }
        return (result * sign);
}
```