Lab 4

Marcus Östmo & Douglas Halse

Task 1

The task was to implement a read buffer of size 16 bytes using C. This should be done using system call and not with buffered I/0 from the standard library. We got some help from our lab assistant and we were informed that we could use the C-function fopen to open the file that we should create the buffer from.

```c
#include <stdio.h>
#include <stdlib.h>

char *p;
char *write_p;
int write_count;

void init_buffer(char size, char** dest)
{
    char* buffer;
    buffer = (char*)malloc(sizeof(char)* size);
    *dest = buffer;
}

char buf_in(FILE * pFile)
{
    if(p[0] == 0)
    {
        int i = 0;
        while(i < 16)
        {
            fread((p + sizeof(char)*i),1,1,pFile);
            i++;
        }
    }
    else
    {
        char temp =  *p;
        p = p + sizeof(char);
        return temp;
    }
}
```

```c
int main()
{
    init_buffer(16, &p);
    init_buffer(16, &write_p);
    FILE * pFile;
    pFile = fopen("text.txt","r");
    if(!pFile)
    {
        printf("Error when reading file");
    }
    else
    {
        buf_in(pFile);

        int i = 0;
        while (i < 16)
        {
            printf("%c", buf_in(pFile));
            i++;
        }
        buf_in(pFile);
        printf("\n");
        i = 0;
        while (i < 16)
        {
            printf("%c", buf_in(pFile));
            i++;
        }
    }
}
```

The first step is to initialize a global pointer to the first element in the buffer. The buffer is then initialized with size 16. Then, the file is opened using fopen in read-mode. The function buf_in is then called which, if and only if the buffers is empty, fills the buffer with 16 characters. If the buffer is not empty buf_in returns the character that the pointer currently points to, and then increments the pointer to the next character.