Question F – Daniyar Nazarbayev [H00204990].

1.

```
fun strip [] = []
| strip (h::t) = if Char.isAlpha(h) then (h)::strip(t)
else strip(t);
```

2.

```
fun next x =
if x=[] then ([],[])
else if Char.isAlpha(hd(x))
then let val (y, z) = next(tl(x))
      in (hd(x)::y, z)
      end
else ([], x);
```

3.

```
fun func x = if x=[] then [] else if Char.isAlpha(hd(x)) then
hd(x)::func(tl(x)) else [];
fun func2 x = if x=[] then [] else if Char.isAlpha(hd(x)) then func2(tl(x))
else tl(x);


fun words x =
if x=[] then []
else
let val y = func(x)
      val z = func2(x)
      in implode(y)::words(z)
      end;
```

4.

```
fun incCount word [] = (word,1)::[]

| incCount word ((stored_word,number)::rest)  = if stored_word=word then
(stored_word,number+1)::rest

else (stored_word, number)::(incCount word rest);
```

5.

```
fun incCount word [] = (word,1)::[]

| incCount word ((stored_word,number)::rest)  = if stored_word=word then
(stored_word,number+1)::rest

else (stored_word, number)::(incCount word rest);


fun counts [] l2 = l2

| counts (hd::tl) l2 = counts(tl) (incCount (hd) (l2));
```

6.

```
fun func x = if x=[] then [] else if Char.isAlpha(hd(x)) then
hd(x)::func(tl(x)) else [];

fun func2 x = if x=[] then [] else if Char.isAlpha(hd(x)) then func2(tl(x))
else tl(x);


fun words x =

if x=[] then []

else

let val y = func(x)

      val z = func2(x)

      in implode(y)::words(z)

      end;


fun incCount word [] = (word,1)::[]

| incCount word ((stored_word,number)::rest)  = if stored_word=word then
(stored_word,number+1)::rest
```

```
else (stored_word, number)::(incCount word rest);


fun counts [] l2 = l2
| counts (hd::tl) l2 = counts(tl) (incCount (hd) (l2));


fun parse file =
counts(words(explode(String.toString(TextIO.inputAll(TextIO.openIn(file))))))
[]
and close file = TextIO.closeIn(file);
```

bolded font is the unique part of the code. The rest are from exercises 3,4,5.

I use TextIO.openIn to open a file, and then take all the input with inputAll, which is of type "vector". Then I convert the result to string and then explode it into a char list. From there I use the function from the 3rd exercise to get a list of strings, and then from 4-5 exercises to count all the words.

I also figured out the purpose of an **and** keyword in SML, which I used to close a file, or at least I hope I did.

Here is an example of a file path = "C:/Users/daniel/Desktop/foo.txt". Use forward slashes.

7.

```
fun display [] = ""
| display ((word,number)::t) = word^" "^Int.toString(number)^"\n"^display(t);
```

give it an input of list with tuples (string * int)

then **print it;**
