Do not turn this page until you have received the signal to start.
(In the meantime, please fill out the identification section above,
and read the instructions below carefully.)

This midterm test consists of 6 questions on 7 pages (including
this one). When you receive the signal to start, please make sure
that your copy of the test is complete. Extra space was left for each
of the programming questions. Please indicate clearly the part of
your work that should be marked.

IMPORTANT: You do not need to include the “#!” line in Bourne
shell programs you are asked to write. In C programs, you do not
need to add “#include” lines, or do error checking unless the question
requires it, or the program would not function correctly given valid
input without error checking.

MARKING GUIDE

# 1: _____/ 3
# 2: _____/ 6
# 3: _____/ 4
# 4: _____/ 5
# 5: _____/ 7
# 6: _____/ 6
TOTAL: _____/31

Good Luck!
Question 1. [3 marks]

Part (a) [1 mark]
Consider the following code snippet:

```c
char str[5];
/* Some code that puts something into str */
printf("Length = %d\n", strlen(str));
```

Is it possible for the print statement to print “Length is 8” without getting any compile time or run time warnings? Why or why not?

Yes. str does not necessarily contain the null termination character.

Part (b) [1 mark]
Explain how a job that has been started in the foreground can be made to run in the background without killing it.

The job can be suspended with $^Z$ and then switched to the background using bg.

Part (c) [1 mark]
My shell script contains the following line:

```bash
if [ $1 == "red" ]
```

But when I run my program I get an error message:

```
[: ==: unary operator expected
```

Explain why the error occurs and modify the code to fix the error.

The error occurs when $1$ is an empty string, so there is no value for the left hand side of the $==$ operator. It can be fixed by putting the $1$ in double quotes.
Question 2. [6 marks]

Given the following code:

```c
char **s = malloc(_________);
char p[10] = "Paul";
char q[10] = "Karen";
char r[10] = "Francois";

*s = p;
*(s+1) = q;
*(s+2) = r;
```

Part (a) [1 mark]
Write the argument to malloc in the statement above so that malloc will allocate just enough space so that
the rest of the code is correct.
Answer: 3 * sizeof(char *)

Part (b) [3 marks]
Give the type of each of the following expressions:

- `&s` (char ***)
- `*s` (char *)
- `**s` (char)
- `s[0]` (char *)
- `&s[1]` (char **)
- `*s[0]` (char)

Part (c) [1 mark]
Fill in the argument to the following printf statement so that the statement prints ‘u’. The argument must
use `s` and not `p` or ‘u’.

```
printf("%c \n", ____________);
```

Answer: `s[0][2]` or `*(s + 1)`

Part (d) [1 mark]
Write one C statement to truncate the string “Francois” so that the following printf statement prints
“Fran”.

```
printf("%s\n", r);
```

Answer: `r[4] = '\0'`;
Question 3. [4 marks]

Running `ls -l` on the current working directory produces the following output:

```
-rw-r--r-- 1 reid reid 20 Oct 18 23:07 cat
-rw-r--r-- 1 reid reid 19 Oct 18 23:08 cow
-rw-r--r-- 1 reid reid 20 Oct 18 23:08 dog
```

The contents of the three files are shown below:

<table>
<thead>
<tr>
<th>cat</th>
<th>cow</th>
<th>dog</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>#!/bin/sh</code></td>
<td><code>#!/bin/sh</code></td>
<td><code>#!/bin/sh</code></td>
</tr>
<tr>
<td><code>echo &quot;hi $1 meow&quot;</code></td>
<td><code>echo &quot;hi $1 moo&quot;</code></td>
<td><code>echo &quot;hi $1 woof&quot;</code></td>
</tr>
</tbody>
</table>

Part (a) [2 marks]

If we run `dog cow` an error message is printed. What is the error message and how do we fix it?

The error message is “Permission denied”. Change the permissions on the dog file so that it is executable: `chmod u+x dog`.

Part (b) [2 marks]

When we execute `cat cow` the output is as follows:

```
#!/bin/sh
echo "hi $1 moo"
```

Explain why and explain how to execute `cat` to produce the output `hi cow meow`.

The system program `cat` is ahead of the current working directory in the `PATH` so it is running that program. Run it as `./cat cow`
Question 4. [5 marks]

Write a Bourne shell program to check the directories in your PATH variable. If an element in the PATH is not a directory or not readable, then print an error message.

To iterate over the PATH you need some way to convert it to a list. Recall that the PATH is a colon-separated list of directories, so it appears in your shell program as a single string. To effectively turn the PATH into a list that you can iterate over, use the following command. The tr command below reads from standard input, replaces every colon in the input with a space, and writes the resulting string to standard output.

```
tr ":" " 
```

```
#!/bin/sh

mypath=`echo $PATH | tr ":" " "`

for p in $mypath
do
    if [ ! -d $p -o ! -r $p ];
    then
        echo $p does not exist or is not readable
    fi
done
```
Question 5.  [7 MARKS]
Following is parts of the man page description of strncpy.

SYNOPSIS
char *strncpy(char *dest, const char *src, size_t n);

DESCRIPTION
The strncpy() function copies the string pointed to by src (including
the terminating ‘\0’ character) to the array pointed to by dest. Not
more than n bytes of src are copied. Thus, if there is no null byte
among the first n bytes of src, the result will not be null-terminated.

In the case where the length of src is less than that of n, the remain-
der of dest will be padded with nulls

RETURN VALUE
The strncpy() function returns a pointer to the destination string dest.

Part (a)  [5 MARKS]
Write strncpy without using any string functions.

char *
mystrncpy(char *dest, char *src, int length)
{
   int i;
   int terminated = 0;
   for(i = 0; i < length; i++) {
      dest[i] = src[i];
      if(src[i] == '\0') {
         break;
         terminated = i;
      }
   }
   if(! terminated) {
      return NULL;
   }

   for(i = terminated; i < length; i++) {
      dest[i] = '\0';
   }
   return dest;
}

Part (b)  [2 MARKS]
Without changing the arguments to mystrncpy, can any errors be caught? If yes, add the error handling
code to your function above, and explain what type of error(s) will be caught. If no, explain why errors
that might occur cannot be handled. (If an error is caught, mystrncpy should return NULL.)

   It is possible to check if src contains a null termination character between 0 and length-1.
Question 6. [6 marks]
Write a Bourne shell program `mygrep` that implements a simple form of grep. The first command line argument is a word to search for, the remaining command line arguments are the names of the files to search in.

You must use a program called `findword` that takes two command line arguments: a word to search for, and a string to look in. It returns 0 if it finds the word in the string and 1 if it does not.

The output of `mygrep` will be one line for each line of every file that contains the word. The line has the format “filename[line number]: line”. For example, if the word “banana” is found on line 2 in the file `fruit`, the output of `mygrep banana fruit` will be `fruit[2]: I like to eat bananas`.

```bash
#!/bin/sh

word=$1
count=0
shift

for file in "$@
  do
    while read line
      do
        count='expr $count + 1'

        ./findword $word "$line"
        if [ $? == 0 ]
          then
            echo "$file[$count]: $line"
        fi
      done < $file
  done
```