SOLUTIONS
Shell Programming—an Introduction

1. This is hello1:

```
#! /bin/sh
echo Hello $@
```

or (here is hello2):

```
#! /bin/sh
echo -n "Hello "
for i
do
    echo -n "$i "
done
echo
```

2. Here is some typical output:

```
$ sh hello1 Nick Urbanik
Hello Nick Urbanik
$ sh -x hello1 Nick Urbanik
+ echo Hello Nick Urbanik
Hello Nick Urbanik
$ sh -x hello2 Nick Urbanik
+ echo -n 'Hello '
Hello + echo -n 'Nick '
Nick + echo -n 'Urbanik '
Urbanik + echo

$ sh -v hello1 Nick Urbanik
#! /bin/sh
echo -n "Hello "
$ sh -v ./hello2 Nick Urbanik
#! /bin/sh
echo -n "Hello "
Hello for i
do
    echo -n "$i "
done
Nick Urbanik echo
```

From the manual page of bash:

-v Print shell input lines as they are read.

-x After expanding each simple command, display the expanded value of `PS4`, followed by the command and its expanded arguments.

`PS4` is a prompt used in tracing execution, which as you can see above, is normally set to "+ ".

3.

```
#! /bin/sh

[ $# -ne 2 ] && echo $0 start finish && exit
start=$1
end=$2
i=$start
while [ $i -le $end ]
do
    echo -n "$i "
    i=$(expr $i + 1)
done
echo
```

The question asks for a `while` loop, so let's use easier arithmetic evaluation notation:

```
#! /bin/sh

[ $# -ne 2 ] && echo $0 start finish && exit
start=$1
end=$2
i=$start
while (( i <= end ))
do
    echo -n "$i "
    (( ++i ))
done
echo
```

Let's see an answer using the alternative `for` loop syntax with arithmetic evaluation:

```
#! /bin/sh

[ $# -ne 2 ] && echo $0 start finish && exit
start=$1
end=$2
for (( i = start; i <= end; ++i ))
do
    echo -n "$i "
done
echo
```

4.

```
#! /bin/sh

for file in *.rpm
```

```
do
    rpm -K $file
    echo
done
```

5.
```
#! /bin/sh

# for file in r*.rpm emacs*.rpm
for file in *.rpm
do
    OUT=$(rpm -K $file) > /dev/null 2>&1
    if echo $OUT | grep -v 'md5 gpg OK' > /dev/null 2>&1
    then
        echo BAD $OUT
    else
        :
        # echo GOOD $OUT
    fi
done
```

6.
```
#! /bin/sh

if ! grep admins /etc/group > /dev/null 2>&1
then
    groupadd admins
done
```