#!/bin/bash
#
# echo prints a given string to stdout.
# Instead of ", try also single quote \\
# and no quotes.
#
echo "Hello World"

age=31
msg="Happy $age-th Birthday! $name"

echo "$msg"
#!/bin/bash
#
# Introducing if statement and grave accent ``.
#
# day=`date +%A`
if test $day == "Monday" −o $day == "Thursday"
then
echo "lecture"
elif test $day == "Friday"
then
echo "lab"
elif test $day == "Tuesday"
then
echo "office hour"
fi

#!/bin/bash
#
# Introducing $# and $1, $2, $3 ..,
# (the command line arguments)
#
if test $# −eq 0; then
    echo "usage: $0 <filename>"
else
    count=`wc −l < $1`
    if test $count −ge 10 ; then
        echo "$1 is too long (longer than 10 lines)"
    fi
fi
#!/bin/bash
#
# Introducing case statement.
# Notice the use of double semicolon.
#
day=`date +%A`
case $day in
  Monday|Thursday)
    echo lecture;;
  Friday)
    echo lab;;
  Tuesday)
    echo office hour;;
  *)
    echo free;;
esac

#!/bin/bash
#
# Introducing for loop.
#
for student in `cat CLASSLIST`
do
  filename=/home/$student/CS2281_LABs/a1/pi.c.pgp
  if test -e $filename; then
    if ! test -e a1/$student; then
      mkdir -p a1/$student
    fi
  cp $filename a1/$student
  else
    echo "WARNING: $student did not submit a1"
  fi
done
#!/bin/bash

# Introducing while loop, and read
sum=0
while read number
do
    sum=$(($sum + $number))
    echo $sum
done

# You can write function in shell script as well. Arguments to functions are accessed with $# and $1, $2, ...

is_odd ()
{
    if test $# == 0; then
        echo "usage: call $0 with 1 argument"
    fi

    remainder=$(($1 % 2))
    if test $remainder == 1; then
        return 0
    else
        return 1
    fi
}

read num
is_odd $num