DAY 12 STAGE 2 REVIEW

Q1. Write a program to print numbers from 1 to 100.

Print fizz if number % 3

Print buzz if number %5

Print fizzbuzz if number %5 & number%3

Ans:

#! /bin/bash

for i in {1..100}

do

if [[ $i%15 -eq 0 ]]

then

echo "FizzBuzz"

elif [[ $i%3 -eq 0 ]]

then

echo "Fizz"

elif [[ $i%5 -eq 0 ]]

then

echo "Buzz"

else

echo $i

fi

done

Q2.Write a program to simulate Rock ,Paper,Scissors in computre. Take input from user and from computer in random.

Ans:

#!/bin/bash

num=$[RANDOM%3 +1]

echo "Choose Your Move"

echo "1. Rock"

echo "2. Scissors"

echo "3. Paper"

read -p "Please select 1-3:" person

case $person in

1)

if [ $num -eq 1 ];then

echo "draw"

elif [ $num -eq 2 ];then

echo "You win"

else

echo "computer win"

fi;;

2)

if [ $num -eq 1 ];then

echo "computer win"

elif [ $num -eq 2 ];then

echo "draw"

else

echo "You win"

fi;;

3)

if [ $num -eq 1 ];then

echo "You win"

elif [ $num -eq 2 ];then

echo "computer win"

else

echo "draw"

fi;;

\*)

echo "must enter a number from 1-3"

esac

Q3. Write a program to generate a series odd,even,and prime from 1 to N. Take input N from user.

Ans:

#! /bin/bash

echo enter m and n

function prime()

{

read m n

for a in $(seq $m $n)

do

k=0

for i in $(seq 2 $(expr $a - 1))

do

if [ $(expr $a % $i) -eq 0 ]

then

k=1

break

fi

done

if [ $k -eq 0 ]

then

echo PrimeNumber=$a

fi

done

}

function oddeven()

{

local i=1

for b in $(seq $m $n)

do

if [ `expr $i % 2` -eq 0 ]

then

echo EvenNumber=$i

else

echo OddNumber=$i

fi

i=`expr $i + 1`

done

}

prime

oddeven

**DAY 3 LINUX COMMANDS ASSIGNMENT**

**Q2. MOVE FILES FROM ONE FOLDER TO THE RESPECTIVE FOLDERS.**

**EXPECTED OUTCOME abc/abc.txt**

**Ans:** #! /bin/bash

for file in $( ls \*.txt)

do

baseName=$( echo $file | awk -F. '{ print $1 }')

echo $file,$baseName

if [ -d $baseName ]

then

rm -r $baseName

fi

mkdir $baseName

cp $file $baseName

Done

**Q3. Append current date to all log files which has extension .log.1 from folder**

**Ans:**

#! /bin/bash

for file in $( ls \*.log.1)

do

baseName=$( echo $file | awk -F. '{ print $1 }')

echo $file,$baseName, $date

current\_time=$(date "+%d.%m.%y")

echo "Current Time : $current\_time"

if [ -d $baseName ]

then

rm -r $baseName

fi

mkdir $baseName

cp $file $baseName

done

**Q4. Archive the files from /var/log folder which have modified 7 days ago and move to your backup folder’**

**Ans:**

Find -mtime7

Mv file.txt backup

**Q5. Print last 4 frequently urls count in sorted order from /var/log/httpd/access.log**

**Ans:** cat access.log | grep http | sort | uniq -c | head -4

**Q6. Print last 4 frequently urls at particular hours /var/log/httpd/access.log**

**Ans:**

find . -type f | grep http | xargs ls -l --time-style="+%d-%m-%Y" | sort | uniq -c | head -4

**Q7. Print list of web response code count in the unique sorted order at specific hours.**

**Ans:**awk '{ print $3 } ' access.log | sort | uniq -c

$3 for third column containing web response codes.

**Q8. Print list of last 10 unique sorted ip from /var/log/httpd/access.log**

**Ans:**grep -Eo '([0-9]{1,3}\.){3}[0-9]{1,3}' access.log | sort | uniq -c | tail -10

**Q9. Check if folder exists or not . If not present, create it.**

**Ans:**

#! /bin/bash

DIR="/c/Users/Vijay/Desktop/minibootcamp/"

if [[ -d "$DIR" ]]

then

echo " $DIR exists "

else

echo " $DIR doesn't exist "

fi

**Q10. Execute command “hello” and “ls” and check execution status**

$ hello

$ ls

$? to check the command execution status.

**Q11. Set environment usersecret=”dH34xJaa23” if already not set.**

**Ans:**

Nano :bash\_profile

Export USER = “ABC”

Export PASSWORD = ”XYZ”

Ctrl + y #to confirm

$source .bash\_profile #to make effect on the changes

**Q12. Create process list table displays provcess id. Parent process id , command name, % of memory consumption, % of cpu utilization.**

**Ans:**

Ps -eo pid,ppid,cmd,%mem,%cpu

**Q13. Data analysis / manipulation (Awk)**

**Operations performed on “data.csv” file.**

**Ans:**

Cat data.csv | grep -i captain | awk ‘{print $1,$2,$3,$7}’ # to print data of column of 1,2,3 and 7 Containing pattern CAPTAIN.

Cat data.csv | grep -I captain | awk ‘{sum+= $7} END {print sum}’ # to calculate sum of column 7 containg pattern CAPTAIN.

Cat data.csv | awk ‘{ if ($3 > 0) {print $0}}’ #print data from all columns having data greater than zero.

Cat data.csv | awk “{if ($7>400000) {print $7}}’ #print total pay greater than 400000

**Q14.Find the difference between original file and and the updated file.**

**Apply changes to original file.**

Mkdir original -> nano original.sh

Mkdir updated ->nano updated.sh

Diff original updated # to find diference between original and updated file

Cp original original-backup #to copy folder original to folder original-backup

Diff updated original-backup #to verify difference between files

DAY 5 LINUX COMMANDS ASSIGNMENT

Q1. Use Random function ((RANDOM)) to generate single digit integer

Ans: echo $(($RANDOM%10))

Q2. Use Random to get dice number integer (1 to 6).

Ans: echo $(($RANDOM%7))

Q3. Add two Random dice number and print the result.

Ans:

#! bin/bash/

a=$(( $RANDOM%7))

b=$(($RANDOM%7))

Result=$( expr $a + $b )

echo "Sum:$result"

Q4.Write a program that reads 5 Random 2 digt values,and then find their sum and average.

Ans: a=$(( RANDOM % (89 ) + 10 ))

b=$(( RANDOM % (89 ) + 10 ))

c=$(( RANDOM % (89 ) + 10 ))

d=$(( RANDOM % (89 ) + 10 ))

e=$(( RANDOM % (89 ) + 10 ))

Sum=$( expr $a + $b + $c + $d + $e )

echo "$Sum"

avg=$( expr $Sum / 5 )

echo "$avg"

Q6.Write a program that reads 5 Random 3 digit values and then outputs the minimum and the maximum value.

Ans:#! /bin/bash

r1=$(( RANDOM%899+100 ));

r2=$(( RANDOM%899+100 ));

r3=$(( RANDOM%899+100 ));

r4=$(( RANDOM%899+100 ));

r5=$(( RANDOM%899+100 ));

if [ $r1 -gt $r2 ] && [ $r1 -gt $r3 ] && [ $r1 -gt $r4 ] && [ $r1 -gt $r5 ]

then

echo " r1 is the maximum value"

elif [ $r2 -gt $r1 ] && [ $r2 -gt $r3 ] && [ $r1 -gt $r4 ] && [ $r1 -gt $r5 ]

then

echo " r2 is the maximum value"

elif [ $r3 -gt $r1 ] && [ $r3 -gt $r2 ] && [ $r3 -gt $r4 ] && [ $r3 -gt $r5 ]

then

echo " r3 is the maximum value"

elif [ $r4 -gt $r1 ] && [ $r4 -gt $r2 ] && [ $r4 -gt $r3 ] && [ $r4 -gt $r5 ]

then

echo " r4 is the maximum value"

else

echo "r5 id the maximum value"

fi

echo "========================================================================================================================================================================"

if [ $r1 -lt $r2 ] && [ $r1 -lt $r3 ] && [ $r1 -lt $r4 ] && [ $r1 -lt $r5 ]

then

echo " r1 is the minimum value"

elif [ $r2 -lt $r1 ] && [ $r2 -lt $r3 ] && [ $r1 -lt $r4 ] && [ $r1 -lt $r5 ]

then

echo " r2 is the minimum value"

elif [ $r3 -lt $r1 ] && [ $r3 -lt $r2 ] && [ $r3 -lt $r4 ] && [ $r3 -lt $r5 ]

then

echo " r3 is the minimum value"

elif [ $r4 -lt $r1 ] && [ $r4 -lt $r2 ] && [ $r4 -lt $r3 ] && [ $r4 -lt $r5 ]

then

echo " r4 is the minimum value"

else

echo "r5 id the minimum value"

fi

Q7. Write a program that takes day and month from the command line and prints true if the day of month is between 20 march and 20 june, false otherwise.

Ans:”

#!/bin/bash

Read -p “Enter the date:” date

Read -p “Enter the month:” month

If [ $month -eq march -a$ date -ge 20 ]

Then

Echo $month $date “true”

Elif [ $month -eq april -a $date -le 30 ]

Then

Echo $month $date “true”

Elif [ $month -eq may -date -le 31 ]

Then

Echo $month $date “true”

Elif [ $month -eq june -date -le 30 ]

Then

Echo $month $date “true”

Else echo “false”

fi

Q8. Write a program to take year as input and output the year as “LeapYear” or “Not a LeapYear”.A Leap Year checks for 4 digit number, divisible by 4,and not by 100 unless divisible by 400.

Ans:

#! /bin/bash

read -y "Enter the year:" year

if [ `expr $year % 400` -eq 0 ]

then

echo leap year

elif [ `expr $year % 100` -eq 0 ]

then

echo not a leap year

elif [ `expr $year % 4` -eq 0 ]

then

echo leap year

else

echo not a leap year

fi

Q9. Write a program to simulate cointoss and print ”heads” and “tails” accordingly.

#! /bin/bash

echo "Welcome to CoinToss"

echo "Choose an option 1.Head 2.Tail"

read -p "Enter your Choice:" choice

coinToss=$(( $RANDOM %2 + 1 ))

if [ $choice -eq $coinToss ]

then

echo " heads"

else

echo "tails"

fi

Q10. Read a single digit number and write the number in word using if ,elif.

Ans: #! bin/bash/

read -p " Enter a number:" num

if [ $num -eq 1 ]

then

echo "one"

elif [ $num -eq 2 ]

then

echo "two"

elif [ $num -eq 3 ]

then

echo "three"

elif [ $num -eq 4 ]

then

echo "four"

else

echo "Out of Range"

Q11. Read a number and display the week day (using if,else)

Ans:

#! /bin/bash

read -p "Enter a Number less than 8:" num

if [ $num -eq 1 ]

then

echo "Sunday"

elif [ $num -eq 2 ]

then

echo "Monday"

elif [ $num -eq 3 ]

then

echo "Tuesday"

elif [ $num -eq 4 ]

then

echo "Wednesday"

elif [ $num -eq 5 ]

then

echo "Thursday"

elif [ $num -eq 6 ]

then

echo "Friday"

elif [$num -eq 7 ]

then

echo "Saturday"

else

echo "Enter a Valid Number"

Fi

Q12. Read a Number 1,10,100,10000 etc and display unit,ten,hundred,………… (using if,else).

Ans:

#! /bin/bash

read -p "Enter a Number in unit,ten,hundred,.... :" num

if [ $num -eq 1 ]

then

echo "Unit"

elif [ $num -eq 10 ]

then

echo "Ten"

elif [ $num -eq 100 ]

then

echo "HUndred"

elif [ $num -eq 1000 ]

then

echo "Thousand"

elif [ $num -eq 10000 ]

then

echo "Ten Thousand"

elif [ $num -eq 100000 ]

then

echo "Hundred Thousand"

elif [ $num -eq 1000000 ]

then

echo "Million"

elif $num -eq 10000000 ]

then

echo "Ten Million"

else

echo "Enter a unit below Ten Millon"

Fi

Q13.Enter 3 numbers and do the following arithmetic operation and find the one that is maximum and minimum.

1. A+b\*c (2) c+a/b (3)a%b+c (4) a\*b+c

#! /bin/bash

a=5

b=8

c=9

r1=$(( $a +$b \* $c ));

r2=$(( $a % $b + $c ));

r3=$(( $c + $a / $b ));

r4=$(( $a \* $b + $c ));

if [ $r1 -gt $r2 ] && [ $r1 -gt $r3 ] && [ $r1 -gt $r4 ]

then

echo " r1 is the maximum value"

elif [ $r2 -gt $r1 ] && [ $r2 -gt $r3 ] && [ $r1 -gt $r4 ]

then

echo " r2 is the maximum value"

elif [ $r3 -gt $r1 ] && [ $r3 -gt $r2 ] && [ $r3 -gt $r4 ]

then

echo " r3 is the maximum value"

else

echo "r4 id the maximum value"

fi

echo "========================================================================================================================================================================"

if [ $r1 -lt $r2 ] && [ $r1 -lt $r3 ] && [ $r1 -lt $r4 ]

then

echo " r1 is the minimum value"

elif [ $r2 -lt $r1 ] && [ $r2 -lt $r3 ] && [ $r1 -lt $r4 ]

then

echo " r2 is the minimum value"

elif [ $r3 -lt $r1 ] && [ $r3 -lt $r2 ] && [ $r3 -lt $r4 ]

then

echo " r3 is the minimum value"

else

echo "r4 id the minimum value"

fi

Q14. Read a single digit number and write the number in word using case.

Ans:

#! bin/bash/

read -p "Enter the number less than 5:" num

case $num in

1) echo "one";;

2) echo "two";;

3) echo "three";;

4) echo "four";;

\*) echo "default";;

esac

1. Read a Number and Display the Week Day.

Ans:

#! bin/bash/

read -p "Enter the number less than 8:" num

case $num in

1) echo "Sunday";;

2) echo "Monday";;

3) echo "Tuesday";;

4) echo "Wednesday";;

5) echo "Thursday";;

6) echo "Friday" ;;

7) echo "Saturday";;

esac

Q16. Read a Number 1,10,100,10000 etc and display unit,ten,hundred,…………

Ans:

#! bin/bash/

read -p "Enter the number for unit , ten , thousand... :" num

case $num in

1) echo "unit";;

10) echo "ten";;

100) echo "hundred";;

1000) echo "thousand";;

10000) echo "ten thousand";;

100000) echo "hundred thousand";;

1000000) echo "million";;

10000000) echo "ten million";;

Esac

Q17. Write a program that takes user input in lenths and does unit conversion of differnet lengths.

Ans:

#! /bin/bash

read -km "Enter distance(in km):" km

meter=`echo $km \\* 1000 | bc`

feet=`echo $meter \\* 3.2808 | bc`

inches=`echo $feet \\* 12 | bc`

cm=`echo $feet \\* 30.48 | bc`

echo "Total meter is : $meter "

echo "Total feet is : $feet "

echo "Total inches is : $inches "

echo "Total centimeters : $cm "

**DAY 6 PRACTICE PROBLEMS**

Q1.Write a program that takes a command line argument n and prints a table of the powers of 2 that are less than or equal to 2^n.

Q3. Write a Program that takes a input and determines a number is prime or not.

Ans: #!/bin/bash

read -p "Enter a number to check:" num

for((i=2; i<=$num/2; i++))

do

# ans=$(( num%i ))

if [ $((num%i)) -eq 0 ]

then

echo "$num is not a prime number."

Else

echo "$num is a prime number."

Fi

done

Q4 Extend the program to take Range as input and display number of prime numbers in Range.

Ans:

#!/bin/bash

Read -p “Enter the highest number:” num

for ((i=1; i<=num; i++))

do

output=$(( $I/2 % 2))

if [ $output -ne 0 ]

then

echo "Prime numbers are: $i"

Fi

Done

Q5.Write a program to compute factorial of a number taken as input.

Ans:

#!/bin/bash

read -p "enter a number for factorial:" num

fact=1

for(( i=1; i<=num; i++ ))

do

fact=$(( $fact\*$i ))

done

echo "$( getfactorial $fact)"

Q6.Write a program to compute factors of a number N using prime factorisation method.

Ans:

#! /bin/bash -x

read -p "Enter an integer greater than one:" num

i=2

count=0

flag=0

for ((i;i<$num;));do

if [ `expr $num % $i` -eq 0 ];then

factor=$i

for ((j=2;j<=`expr $factor / 2`;));do

flag=0

if [ `expr $factor % $j` -eq 0 ];then

flag=1

break

fi

j=`expr $j + 1`

done

if [ $flag -eq 0 ];then

echo "[ $factor ]"

count=1

fi

fi

i=`expr $i + 1`

done

if [ $count -eq 0 ];then

echo "no prime factors found except $num"

fi

Q7.Write a program that takes command line argument n and prints a table of the powers of 2 that are less than or equal to 2^n till 256 is reached.

Ans: # /bin/bash -x

i=2

while [ $i -le 256 ]

do

echo $i

i=$((2\*i))

done

Q8.Find the magic number

1. Ask the user to think a number n between 1 to 100
2. Then check with the user if the number is less than n/2 or greater.
3. Repeat till the magic number is reached.

Ans:

Q9. Extend cointoss problem till heads or tails win 11 times.

Ans:

#! /bin/bash

MAX\_HEADS=11

MAX\_TAILS=11

totalHeads=0

totalTails=0

while [ $totalHeads -le 11 ] && [ $totalTails -le 11 ]

do

echo "Welcome to CoinToss"

coinToss=$(( $RANDOM %2 + 1 ))

echo "Choose an option 1.Head 2.Tail"

read -p "Enter your Choice:" choice

if [ $choice -eq $coinToss ]

then

echo " heads"

else

echo "tails"

fi

done

Q11. Program to convert temperature from farenheight to celcious and vice-versa.

Ans:

function myfunct(){

echo $1

}

**echo** "1. Convert Celsius temperature into Fahrenheit"

**echo** "2. Convert Fahrenheit temperatures into Celsius"

**echo** -n "Select your choice (1-2) : "

**read** choice

**if** **[** $choice -eq 1 **]then**

**echo** -n "Enter temperature (C) : "

**read** tc

*# formula Tf=(9/5)\*Tc+32*

tf=$**(echo** "scale=2;((9/5) \* $tc) + 32" **|bc)**

**echo** "$tc C = $tf F"**elif** **[** $choice -eq 2 **]then**

**echo** -n "Enter temperature (F) : "

**read** tf

*# formula Tc=(5/9)\*(Tf-32)*

tc=$**(echo** "scale=2;(5/9)\*($tf-32)"**|bc)**

**echo** "$tf = $tc"**else**

**echo** "Please select 1 or 2 only"

**exit** 1

**fi**

Q12. Write a function to check if two numbers are palindromes.

Ans:

#! /bin/bash -x

read -p "Enter first number to check:" n

#n=12321

read -p "Enter second number to check:" rev

rev=$( echo $n | rev)

if [ $n -eq $rev ];

then

echo "Number is palindrome"

else

echo "Number is not palindrome"

fi

Q13.(a) Write a function to check a number is prime or not

Ans:

function prime()

{

for((i=2; i<=num; i++))

do

if [ `expr $num % $i` == 0 ]

then

echo $num is not prime

exit

fi

done

echo $num is a prime number

}

read num;

prime "$num";

(b) Write a function to check the number is pelindrome or not.

Ans:

Function checkpalindrome()

{

Local s =$1

For n in $s;

Do

While [ $n -gt 0 ]

rev=$( echo $n | rev)

if [ $n -eq $rev ];

then

echo "Number is palindrome"

else

echo "Number is not palindrome"

fi

}

DAY 7 ARRAY PRACTICE PROBLEMS

Q1.(a)Write a program that generates 10 random 3 digits number.

(b)Store this random number in an array.

(c)Find second largest and second smallest number of array.

Ans:

#! /bin/bash -x

a=$(( RANDOM % (899 ) + 100 ))

b=$(( RANDOM % (899 ) + 100 ))

c=$(( RANDOM % (899 ) + 100 ))

d=$(( RANDOM % (899 ) + 100 ))

e=$(( RANDOM % (899 ) + 100 ))

f=$(( RANDOM % (899 ) + 100 ))

g=$(( RANDOM % (899 ) + 100 ))

h=$(( RANDOM % (899 ) + 100 ))

i=$(( RANDOM % (899 ) + 100 ))

j=$(( RANDOM % (899 ) + 100 ))

declare -a digits

digits=("$a" "$b" "$c" "$d" "$e" "$f" "$g" "$h" "$i" "$j")

echo ${digits[\*]}

Q2.Extend above program to find 2nd smallest and 2nd gighest number using sort method.

Ans:

#! /bin/bash -x

a=$(( RANDOM % (899 ) + 100 ))

b=$(( RANDOM % (899 ) + 100 ))

c=$(( RANDOM % (899 ) + 100 ))

d=$(( RANDOM % (899 ) + 100 ))

e=$(( RANDOM % (899 ) + 100 ))

f=$(( RANDOM % (899 ) + 100 ))

g=$(( RANDOM % (899 ) + 100 ))

h=$(( RANDOM % (899 ) + 100 ))

i=$(( RANDOM % (899 ) + 100 ))

j=$(( RANDOM % (899 ) + 100 ))

declare -a digits

digits=("$a" "$b" "$c" "$d" "$e" "$f" "$g" "$h" "$i" "$j")

echo ${digits[\*]}

for ((i = 0; i<10; i++))

do

for((j = 0; j<10-i-1; j++))

do

if [ ${digits[j]} -gt ${digits[$((j+1))]} ]

then

# swap

temp=${digits[j]}

digits[$j]=${digits[$((j+1))]}

digits[$((j+1))]=$temp

fi

done

done

echo "Array in sorted order :"

echo ${digits[\*]}

echo "Second smallest Number"

echo ${digits[1]}

echo "Second largest Number"

echo ${digits[8]}

Q3.Extrend prime factorisation program to store all the prime factors of number n in an array and display output.

Ans:

#! /bin/bash -x

declare -a factor

read -p "Enter an integer greater than one:" num

i=2

count=0

flag=0”

for ((i;i<$num;));do

if [ `expr $num % $i` -eq 0 ];then

factor=$i

for ((j=2;j<=`expr $factor / 2`;));do

flag=0

if [ `expr $factor % $j` -eq 0 ];then

flag=1

break

fi

j=`expr $j + 1`

done

if [ $flag -eq 0 ];then

echo "[ $factor ]"

count=1

fi

fi

i=`expr $i + 1`

done

if [ $count -eq 0 ];then

echo "no prime factors found except $num"

fi

echo ${factor[\*]}

Q4. Write a program to show sum of 3 digits integer adds to zero.

Ans:

#! /bin/bash

declare -a digits

digits=(5 4 3 -2 -3)

sum=`expr 5 - 2 - 3`

echo ${digits[\*]}

echo $sum: 5 -2 -3

Q5. Take a number from range 0 to 100 ,and find that digits that are repeated twice like 33 and 77 and store them in an array.

Ans: #! /bin/bash -x

#read -p "Enter a number: " number

number=11

i=1

while [ $i -le 9 ]

do

echo "$((number\*i))"

i=$((i+1))

done

DAY 8 ASSIGNMENT DICTIONARY

Q1.Write a program in the following steps

1. Roll a dice and find the number between 1 and 6.
2. Repeat the dice roll and find the result each time
3. Store the result in a dictionary
4. Repeat till one of the the member has reached 10 times.
5. Find the number that reached maximum times and one that was for minimum times.

Ans:

#!/bin/bash

echo "Welcome to DiceRoll:" DiceRoll

read DiceRoll

for ((j;j <= DiceRoll; j++))

do

randomNumber=0

#(this is for initialize the number)

randomNumber=$((RANDOM%(7+1-1)))

DICEROLL+=($randomNumber)

done

echo "${DICEROLL[@]}"

for ((i = 0; i<DiceRoll; i++))

do

for((j = 0; j<DiceRoll-i; j++))

do

if [ ${DICEROLL[j]} -gt ${DICEROLL[$((j+1))]} ]

then

temp=${DICEROLL[j]}

DICEROLL[$j]=${DICEROLL[$((j+1))]}

DICEROLL[$((j+1))]=$temp

fi

count=0

for randomNumber in ${DICEROLL[\*]};

do

if [[ $randomNumber = 6 ]]; then

(( count++ ))

fi

done

done

done

echo $count

echo "Diceroll results in sorted order :"

echo ${DICEROLL[\*]}

Q2. Write a program to genenrate a birth month of 50 individuals between the year 92 and 93.Find all the individuals having birthdays in the same month.Store it to finnaly print.

Ans:

echo -n "enter the number of people:";

read people

for ((j;j <= people; j++))

do

randomNumber=0

randomNumber=$((1 + RANDOM%(12-1+1)))

BIRTHMONTH+=($randomNumber)

done

echo "${BIRTHMONTH[@]}"

for ((i = 0; i<50; i++))

do

for((j = 0; j<50-i-1; j++))

do

if [ ${BIRTHMONTH[j]} -gt ${BIRTHMONTH[$((j+1))]} ]

then

temp=${BIRTHMONTH[j]}

BIRTHMONTH[$j]=${BIRTHMONTH[$((j+1))]}

BIRTHMONTH[$((j+1))]=$temp

fi

done

done

echo "Array in sorted order :"

echo ${BIRTHMONTH[\*]}

case $BIRTHMONTH in

1) echo "Birthmonth is January";;

2) echo "Birth month is February";;

3) echo "Birthmonth is March";;

4) echo "Birthmonth is April";;

5) echo "Birthmonth is May";;

6) echo "Birthmonth is June";;

7) echo "Birthmonth is July";;

8) echo "Birthmonth is August";;

9) echo "Birthmonth is September";;

10) echo "Birthmonth is October";;

11) echo "Birthmonth is November";;

\*) echo "Birthmonth is December";;

esac