Requirements:

Create a list of random ints and track the time to sort them using different algorithms; bubble and/or insertion. Have an interface the user can use to run the program. Needs to be modular, so create functions to: create the list, perform a bubble sort, perform an insertion sort, track the time taken for a sort to happen, generate a user interface.

Proposed components:

For function to generate random int list:

- Takes parameter length to define how long the list is: positive integer
- Use random module to generate a list of random numbers
- Return the list

For function to perform a bubble sort:

- Takes parameter list
- swaps=false
- Repeat this until there are no swaps left, or the iteration amount is the length of the list:
- For all items in the list
- If the current item selected in the list is the last item, break,
- Otherwise, if it is greater than the item next to it, swap them and set swaps to true
- End repeat
- Return list

For function to perform an insertion sort:

- Takes parameter list
- Create a new list
- For every item in the list
- Add the item to the new list
- Sort the new list
- When finished, return the new list

For function to get the time taken:

- Takes parameters list and sort
- Gets the current time
- Performs the passed sort
- Return the difference between the time now and the start time

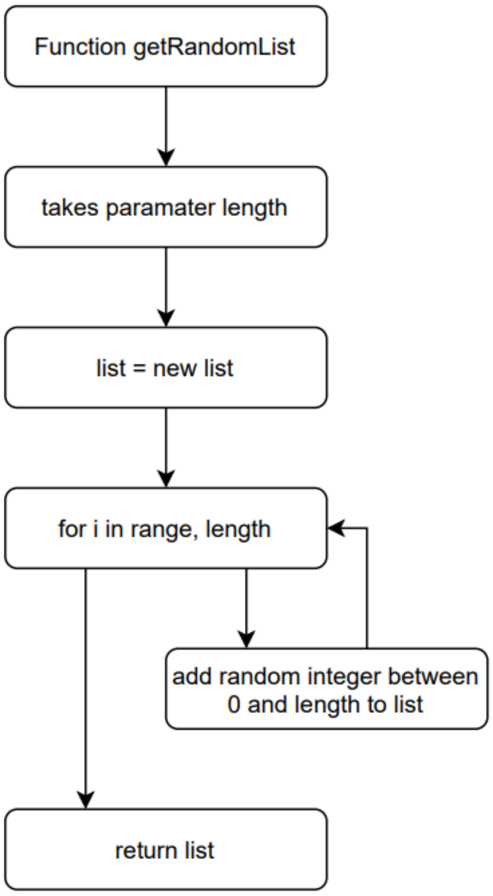
For function to generate a menu:

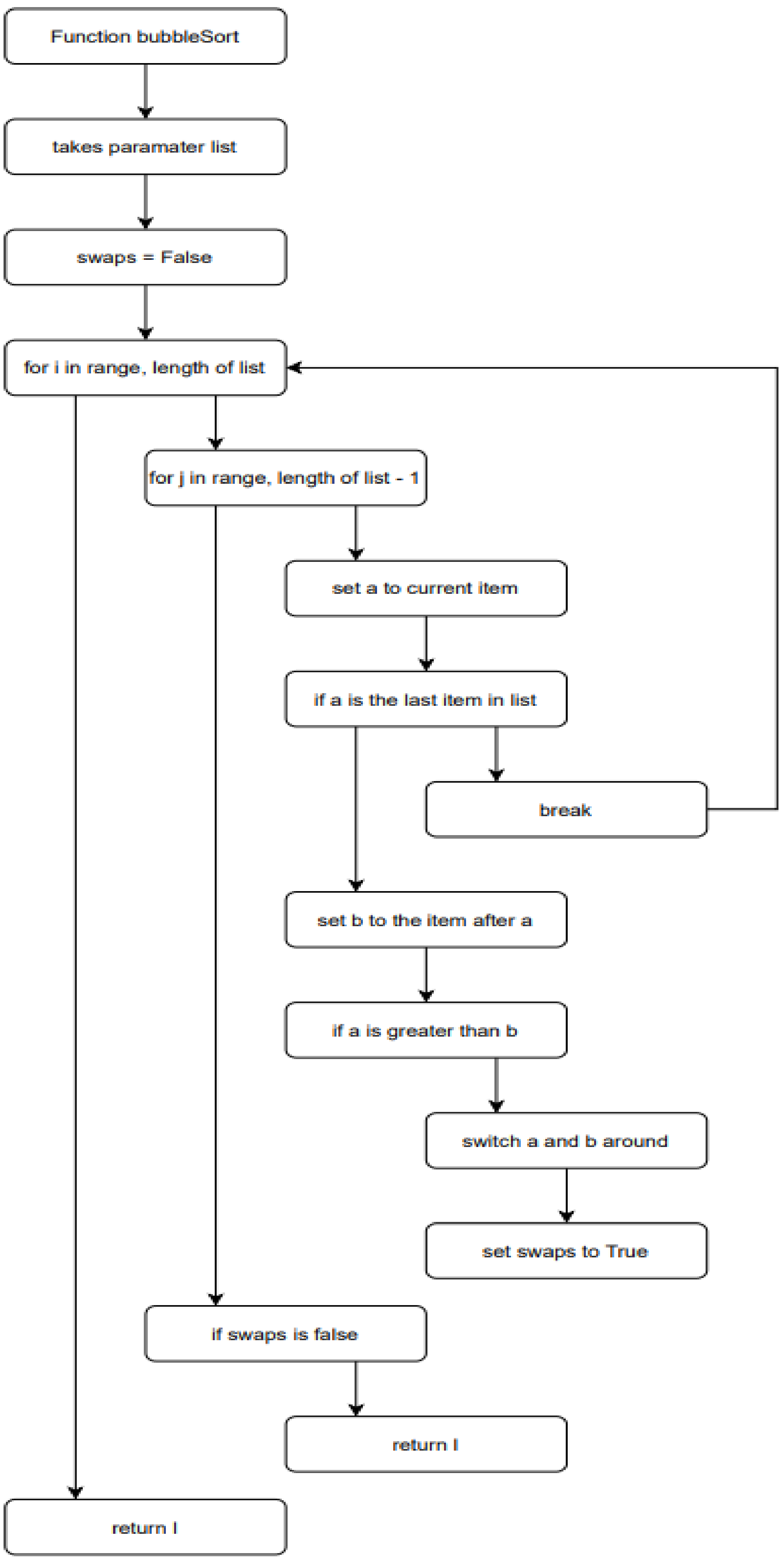
- At any point, if the user enters “exit” in any case, it will exit the program. The user will be told this at the start
- The user will be asked how long they want the list to be.
- The user will then be asked which sort they would like to test

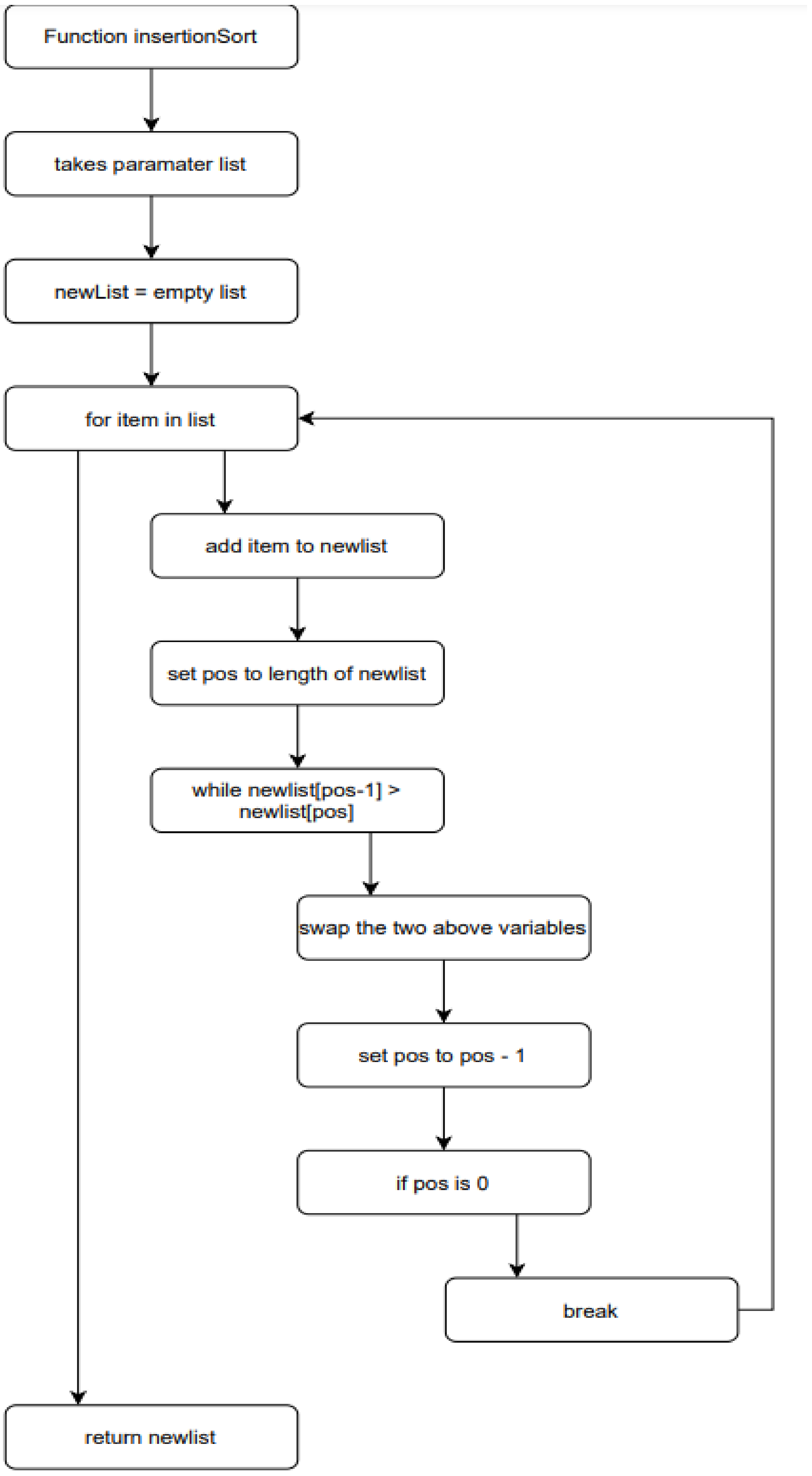
- (All user inputs will have the appropriate error checking on them)
- The program will then execute the sort on a copy of the generated list, and inform the user of the results
- The program will then ask the user whether they would like to perform the other sort on the same list and compare the times
- The program will then perform the opposite search, if the user says yes,  and output the results
- The reason the sorts are done on a copy of the generated list is to allow for multiple different sorts to be tested.
- The user will then be asked whether they would like to start again.
- The program runs indefinitely until the user enters exit, or no to the above question, at which point the program will exit.

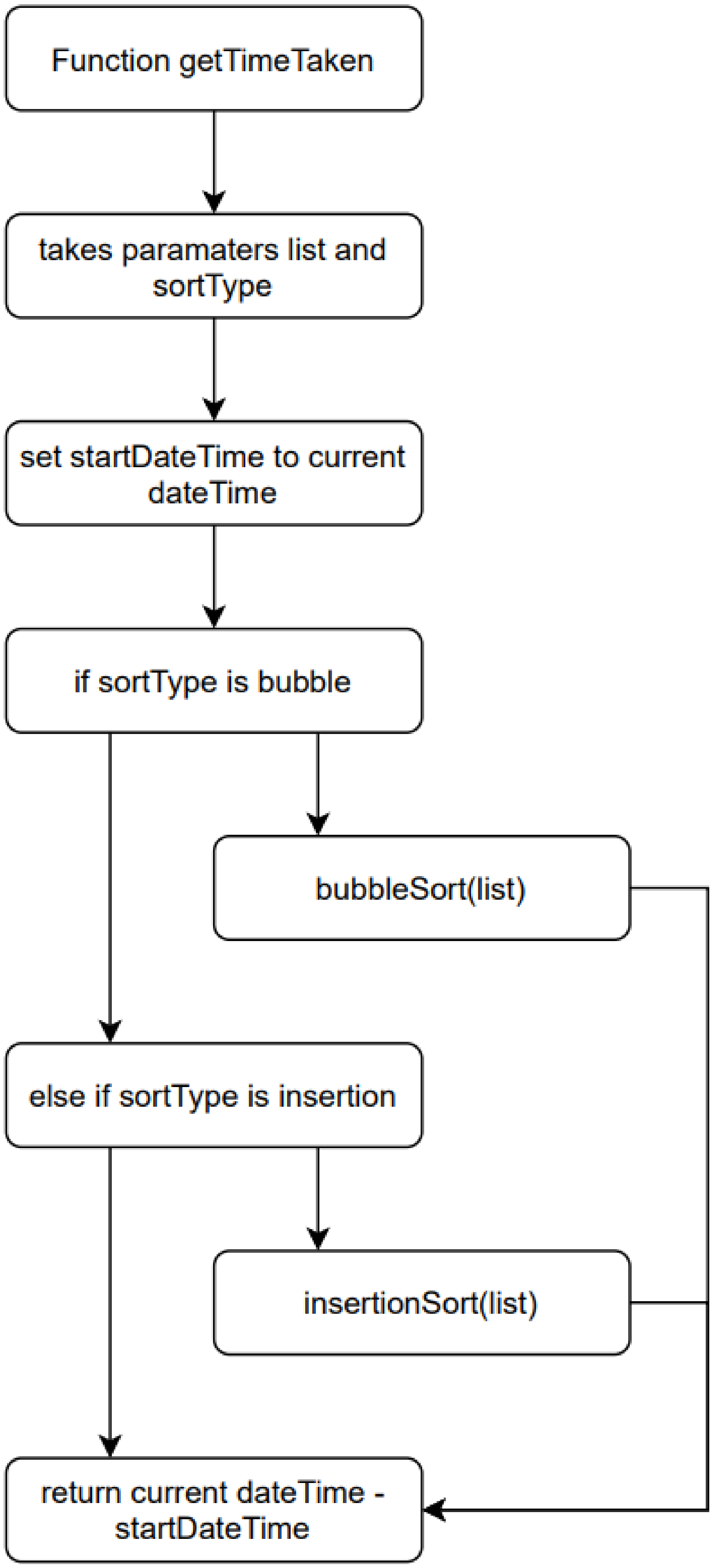
The program will run (i.e. call the function to display the menu) if this is the file that was executed.

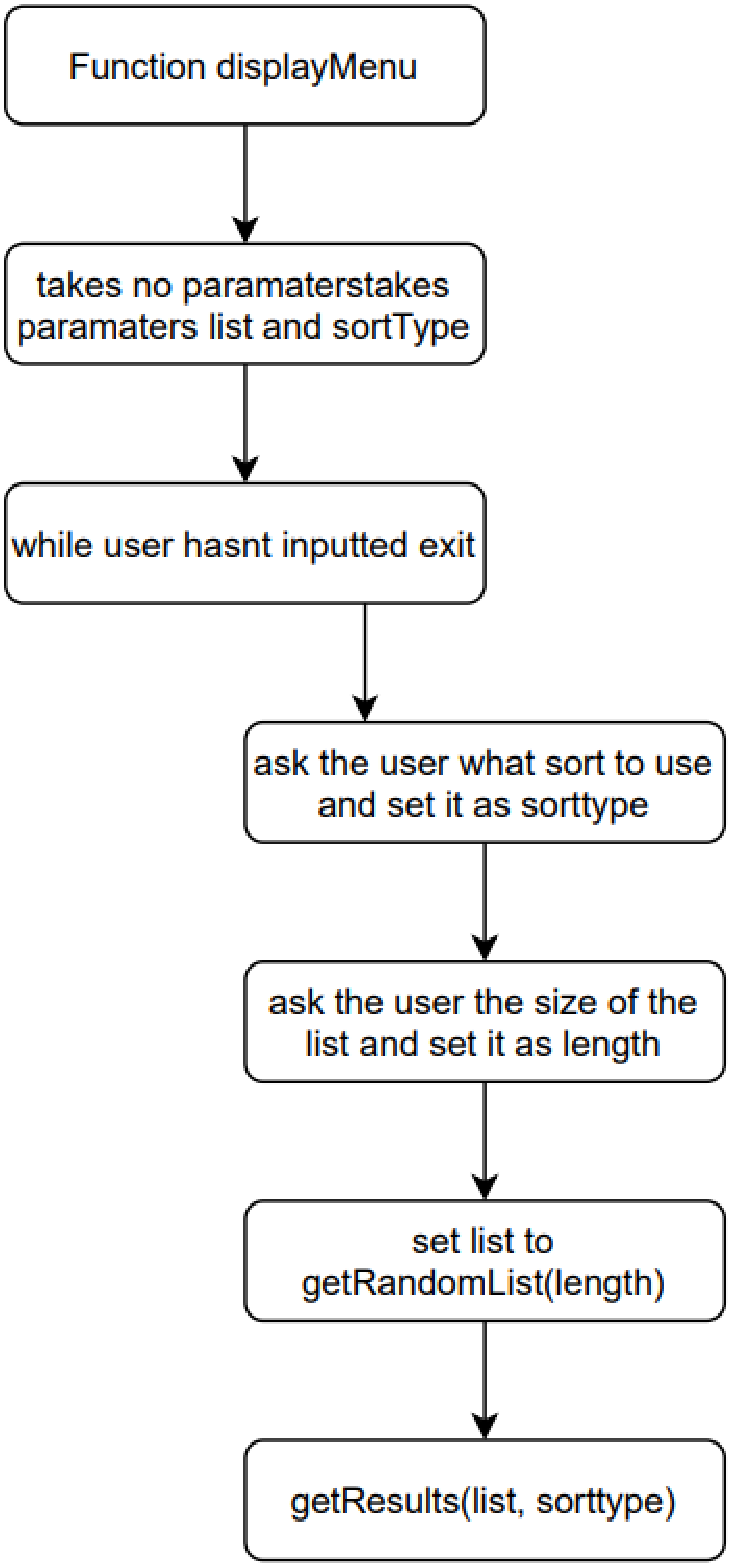
Flowcharts for functions:




Function bubbleSort
takes paramater list
swaps = False
for i in range, length of list
for j in range, length of list - 1
set a to current item
if a is the last item in list
break
set b to the item after a
if a is greater than b
switch a and b around
set swaps to True
if swaps is false
return l
return l

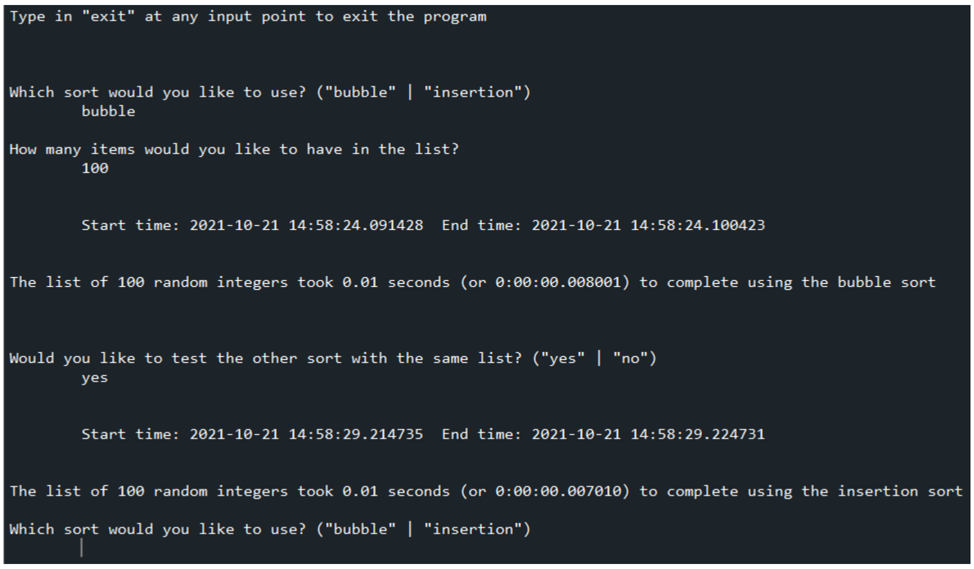

Function insertionSort
takes paramater list
newList = empty list
for item in list
add item to newlist
set pos to length of newlist
while newlist[pos-1] > newlist[pos]
swap the two above variables
set pos to pos - 1
if pos is 0
break
return newlist


Function getTimeTaken
takes paramaters list and sortType
set startDateTime to current dateTime
if sortType is bubble
bubbleSort(list)
else if sortType is insertion
insertionSort(list)
return current dateTime - startDateTime


Function displayMenu
takes no paramaterstakes paramaters list and sortType
while user hasnt inputted exit
ask the user what sort to use and set it as sorttype
ask the user the size of the list and set it as length
set list to getRandomList(length)
getResults(list, sorttype)

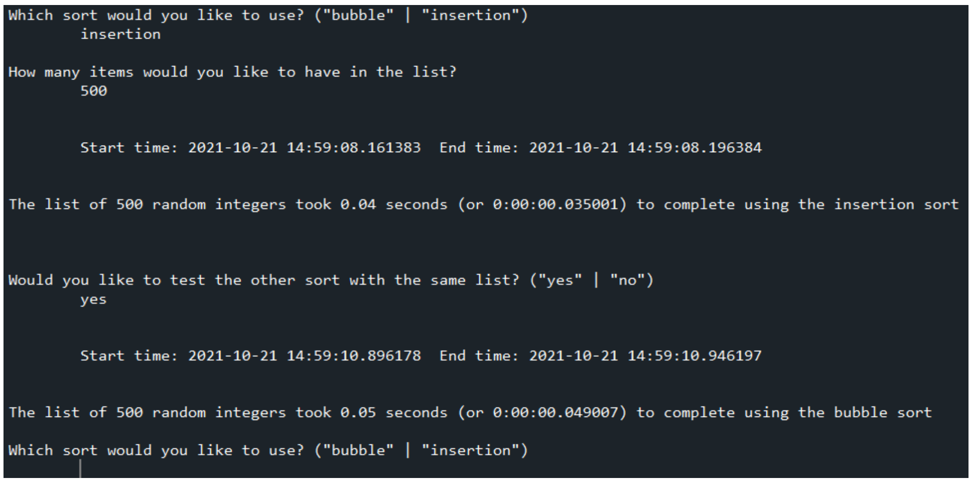
Testing results:

## 100 elements



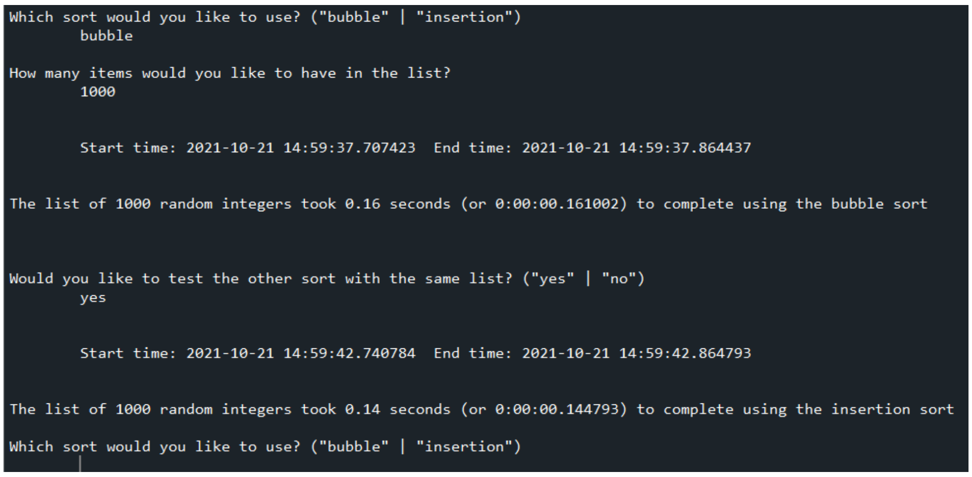
```
Type in "exit" at any input point to exit the program


Which sort would you like to use? ("bubble" | "insertion")
        bubble

How many items would you like to have in the list?
        100


        Start time: 2021-10-21 14:58:24.091428  End time: 2021-10-21 14:58:24.100423


The list of 100 random integers took 0.01 seconds (or 0:00:00.008001) to complete using the bubble sort



Would you like to test the other sort with the same list? ("yes" | "no")
        yes


        Start time: 2021-10-21 14:58:29.214735  End time: 2021-10-21 14:58:29.224731


The list of 100 random integers took 0.01 seconds (or 0:00:00.007010) to complete using the insertion sort

Which sort would you like to use? ("bubble" | "insertion")
```

## 500 elements



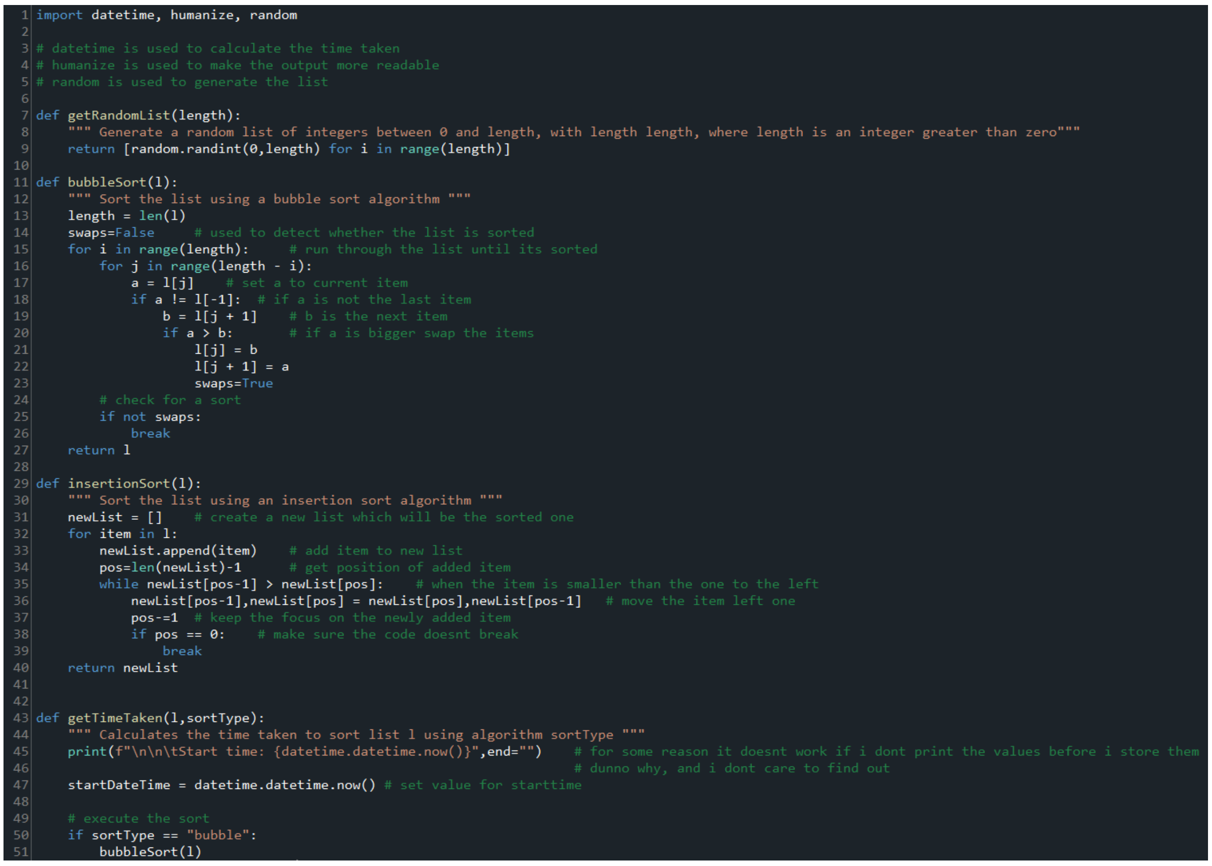
```
Which sort would you like to use? ("bubble" | "insertion")
        insertion

How many items would you like to have in the list?
        500


        Start time: 2021-10-21 14:59:08.161383  End time: 2021-10-21 14:59:08.196384


The list of 500 random integers took 0.04 seconds (or 0:00:00.035001) to complete using the insertion sort



Would you like to test the other sort with the same list? ("yes" | "no")
        yes


        Start time: 2021-10-21 14:59:10.896178  End time: 2021-10-21 14:59:10.946197


The list of 500 random integers took 0.05 seconds (or 0:00:00.049007) to complete using the bubble sort

Which sort would you like to use? ("bubble" | "insertion")
```

## 1000 elements

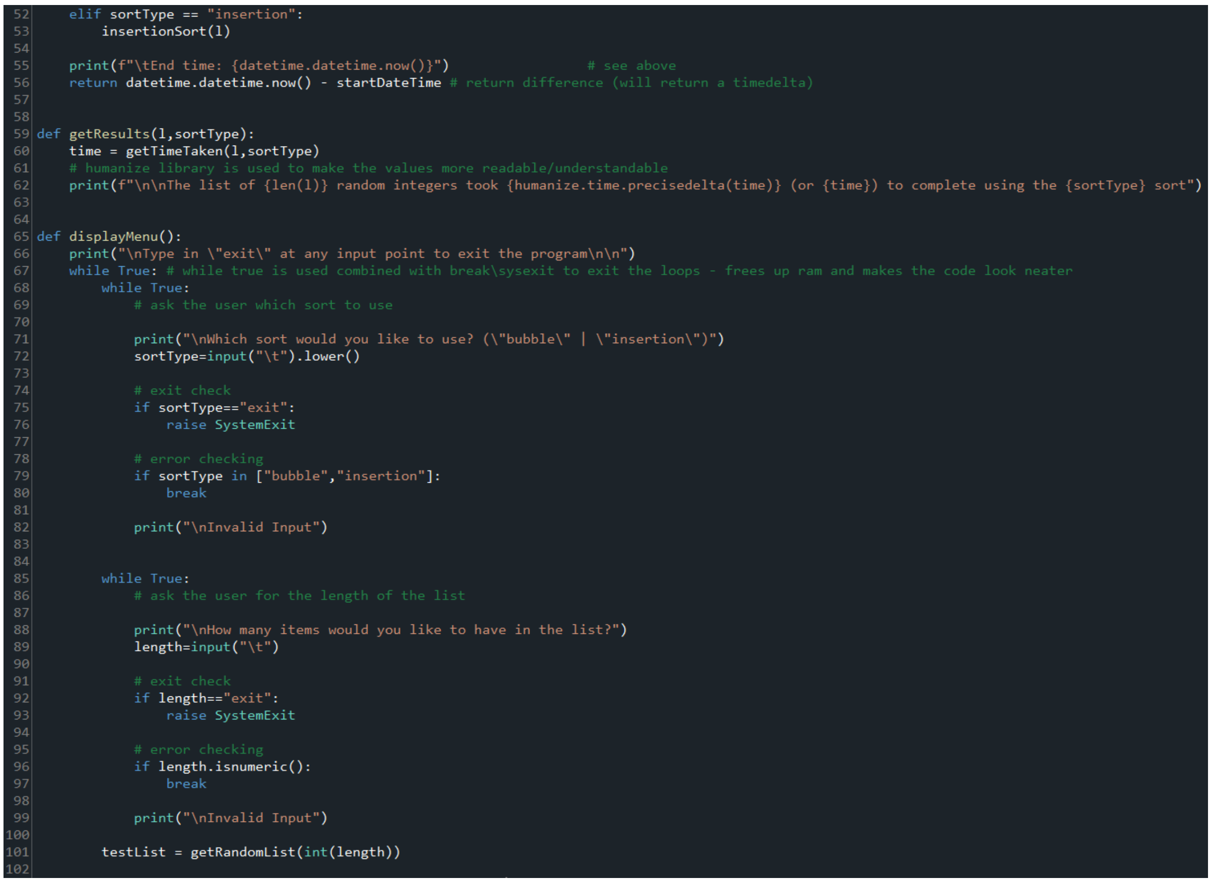


```
Which sort would you like to use? ("bubble" | "insertion")
        bubble

How many items would you like to have in the list?
        1000


        Start time: 2021-10-21 14:59:37.707423  End time: 2021-10-21 14:59:37.864437


The list of 1000 random integers took 0.16 seconds (or 0:00:00.161002) to complete using the bubble sort



Would you like to test the other sort with the same list? ("yes" | "no")
        yes


        Start time: 2021-10-21 14:59:42.740784  End time: 2021-10-21 14:59:42.864793


The list of 1000 random integers took 0.14 seconds (or 0:00:00.144793) to complete using the insertion sort

Which sort would you like to use? ("bubble" | "insertion")
```

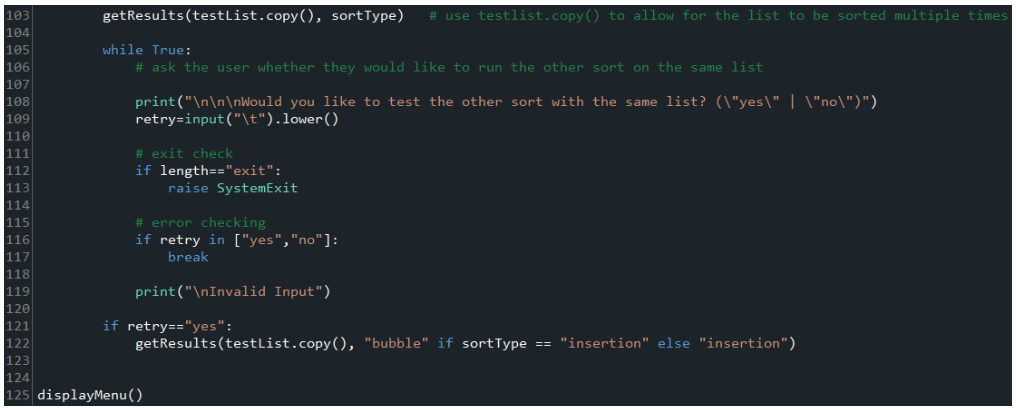
Screenshots of code:



```python
import datetime, humanize, random

# datetime is used to calculate the time taken
# humanize is used to make the output more readable
# random is used to generate the list

def getRandomList(length):
    """ Generate a random list of integers between 0 and length, with length length, where length is an integer greater than zero"""
    return [random.randint(0,length) for i in range(length)]

def bubbleSort(l):
    """ Sort the list using a bubble sort algorithm """
    length = len(l)
    swaps=False     # used to detect whether the list is sorted
    for i in range(length):     # run through the list until its sorted
        for j in range(length - i):
            a = l[j]    # set a to current item
            if a != l[-1]:  # if a is not the last item
                b = l[j + 1]    # b is the next item
                if a > b:       # if a is bigger swap the items
                    l[j] = b
                    l[j + 1] = a
                    swaps=True
        # check for a sort
        if not swaps:
            break
    return l

def insertionSort(l):
    """ Sort the list using an insertion sort algorithm """
    newList = []    # create a new list which will be the sorted one
    for item in l:
        newList.append(item)    # add item to new list
        pos=len(newList)-1      # get position of added item
        while newList[pos-1] > newList[pos]:    # when the item is smaller than the one to the left
            newList[pos-1],newList[pos] = newList[pos],newList[pos-1]   # move the item left one
            pos-=1  # keep the focus on the newly added item
            if pos == 0:    # make sure the code doesnt break
                break
    return newList


def getTimeTaken(l,sortType):
    """ Calculates the time taken to sort list l using algorithm sortType """
    print(f"\n\n\tStart time: {datetime.datetime.now()}",end="")    # for some reason it doesnt work if i dont print the values before i store them
                                                                    # dunno why, and i dont care to find out
    startDateTime = datetime.datetime.now() # set value for starttime

    # execute the sort
    if sortType == "bubble":
        bubbleSort(l)
```



```python
    elif sortType == "insertion":
        insertionSort(l)

    print(f"\tEnd time: {datetime.datetime.now()}")                   # see above
    return datetime.datetime.now() - startDateTime # return difference (will return a timedelta)


def getResults(l,sortType):
    time = getTimeTaken(l,sortType)
    # humanize library is used to make the values more readable/understandable
    print(f"\n\nThe list of {len(l)} random integers took {humanize.time.precisedelta(time)} (or {time}) to complete using the {sortType} sort")


def displayMenu():
    print("\nType in \"exit\" at any input point to exit the program\n\n")
    while True: # while true is used combined with break\sysexit to exit the loops - frees up ram and makes the code look neater
        while True:
            # ask the user which sort to use

            print("\nWhich sort would you like to use? (\"bubble\" | \"insertion\")")
            sortType=input("\t").lower()

            # exit check
            if sortType=="exit":
                raise SystemExit

            # error checking
            if sortType in ["bubble","insertion"]:
                break

            print("\nInvalid Input")


        while True:
            # ask the user for the length of the list

            print("\nHow many items would you like to have in the list?")
            length=input("\t")

            # exit check
            if length=="exit":
                raise SystemExit

            # error checking
            if length.isnumeric():
                break

            print("\nInvalid Input")

        testList = getRandomList(int(length))

```



```python
        getResults(testList.copy(), sortType)   # use testlist.copy() to allow for the list to be sorted multiple times

        while True:
            # ask the user whether they would like to run the other sort on the same list

            print("\n\n\nWould you like to test the other sort with the same list? (\"yes\" | \"no\")")
            retry=input("\t").lower()

            # exit check
            if length=="exit":
                raise SystemExit

            # error checking
            if retry in ["yes","no"]:
                break

            print("\nInvalid Input")

        if retry=="yes":
            getResults(testList.copy(), "bubble" if sortType == "insertion" else "insertion")


displayMenu()
```