Introduction to Programming

UNIVERSITY OF CAPE TOWN
IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

Testing and Debugging

Errors

CSC1

Hussein Suleman
Department of Computer Science

Errors

Hussein Suleman
Department of Computer Science
University of Cape Town

Problem

Write a program that reads in a list of **test scores** and **classifies** them into ranks:

1, 2+, 2-, 3, FS, F

and, for each rank, displays:

- the number of students in the rank
- (output as a **histogram**).

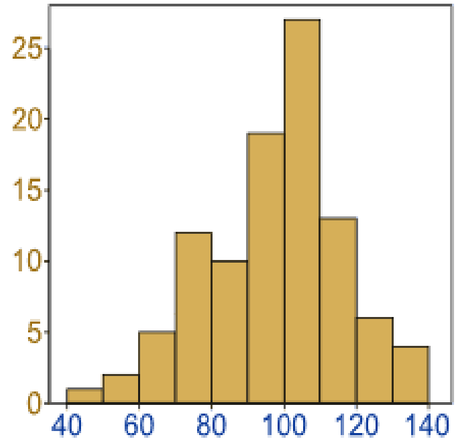
Problem

Write a program that reads in a list of **test scores** and **classifies** them into ranks:

1, 2+, 2-, 3, FS, F

and, for each rank, displays:

- the number of students in the rank
- (output as a **histogram**).



Histogram:

A graphical display where the data is grouped into ranges (such as "100 to 149", "150 to 199", etc), and then plotted as bars. Similar to a Bar Graph, but in a **Histogram** each bar is for a range of data.

markClassificationSkeleton.py

```
def main():
    i,Fi,uS,S,Th,fS,Fa=0,0,0,0,0,0,0 #what am I planning to do with these?
    val = input("Enter next test score #"+str(i)+" (press 'Enter' key to stop):")
    while val!='': #sentinal loop to read in unspecified no. of test scores
        i+=1
        # TODO: add in code to count number in each category
        val =input("Enter next test score #"+str(i)+" (press 'Enter' key to stop):")
    histogram(Fi,uS,S,Th,fS,Fa)
    print('='*10) #Prints line to indicate end of program

def histogram(Fi,uS,S,Th,fS,Fa):
    print("<<< Insert histogram here >>>")

if __name__ == '__main__':  #what does this do?
    main()
```

markClassificationSkeleton.py

- How do we demonstrate that the completed program is correct?

Testing and Debugging

- **Test:**
  - Check if there are errors.
  - Demonstrate that no errors have been found.
- **Debug:**
  - Find the cause of a known error.
  - Repair the code.

Errors and testing: Quick Poll

In a typical hour spent programming, how many minutes do you spend fixing errors?

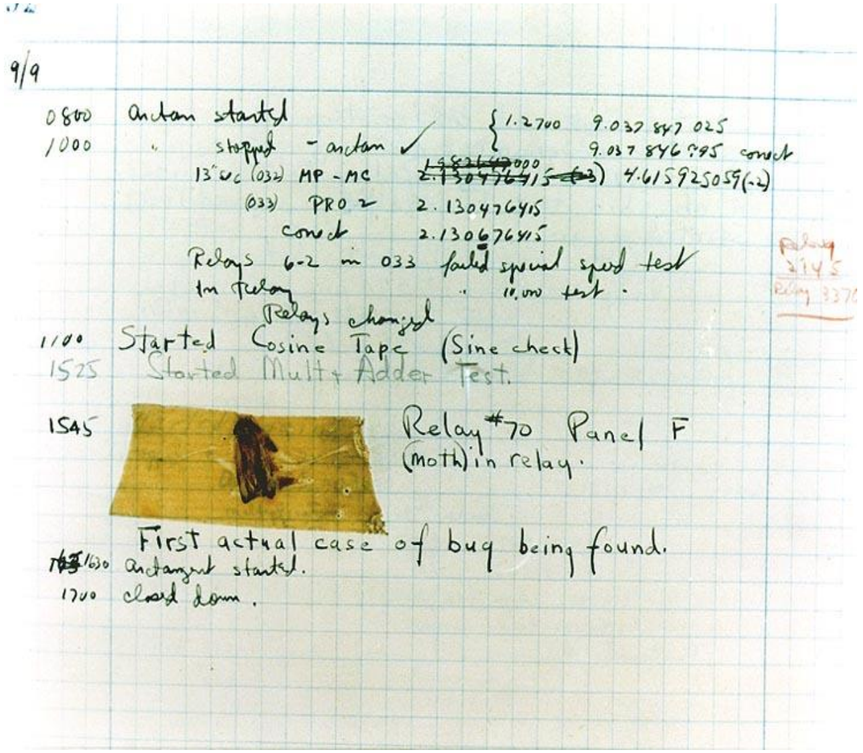
Errors

- What is an error?
  - When your program does not behave as intended or expected.
- What is a bug?
  - “…there is a bug in my program …”
- Debugging
  - the art of removing errors

Errors and testing: Quick Poll

- In a typical hour spent programming, how many minutes do you spend fixing errors? 45+?
- Errors are unavoidable, even for the best programmers.
- Aim to program so that debugging time is reduced
  - for yourself.
  - and others in the future.

The First “Bug”?



The moth was found by Grace Hopper

- Rear Admiral Grace Murray Hopper (December 9, 1906 - January 1, 1992) was an American computer scientist and naval officer
- Worked on Mark I at Harvard
- The first USA computer science "Man of the Year" in 1969.

Reference: http://en.wikipedia.org/wiki/Grace_Hopper

The tale is that the original 'bug' was a moth, which caused a hardware fault in the Harvard Mark I.

Interview on Letterman show
https://www.youtube.com/watch?v=1vcErOPofQ&list=PLBFD1BAAAD218D3EB&index=180

Types of Errors – When (1)

- “Compile”-time Error
  - Discovered when program is checked by the Python interpreter, before it is run.
  - A result of improper use of Python language.
    - usually Syntax Errors.
    - e.g. product = x y

Types of Errors – When (2)

- Run-time Error
  - Program structure is correct, but does not execute as expected.

    e.g.

    ```
    x = 0
    y = 15/x
    ```

- Examples of Python runtime errors:
  - division by zero
  - performing an operation on incompatible types
  - using an identifier that has not been defined

Types of Errors – Why (1)

- Syntax Error
  - Program does not pass checking/compiling stage.
  - Improper use of Python language.
    - e.g. product = x y

Syntax errors are analogous to spelling or grammar mistakes in a language like English:
e.g. “Would you some tea?”

does not make sense – it is missing a verb.

Types of Errors – Why (1)

- Common Python syntax errors:
  - leaving out a keyword
  - putting a keyword in the wrong place
  - leaving out a symbol, such as a colon, comma or brackets
  - misspelling a keyword
  - incorrect indentation

Types of Errors – Why (2)

- Logic Error
  - Program passes checking/compiling and runs but produces incorrect results or no results - because of a flaw in the algorithm or implementation of algorithm.
    - e.g. product = x + y

Poll

What kind of error is in this program?

```
def Maximum(x,y):
    z=x
    if (x<y): z==y
    return z
```

- A – `compile-time, syntax`
- B – `compile-time, logic`
- C – `runtime, syntax`
- D – `runtime, logic`

Solution

What kind of error is in this program?

```
def Maximum(x,y):
    z=x
    if (x<y): z==y
    return z
```

- A – `compile-time, syntax` 
- B – `compile-time, logic`
- C – `runtime, syntax`
- D – `runtime, logic`

Poll

What kind of error is in this program?

```
def Maximum(x,y):
    z=x
    if (x>y): z=y
    return z
```

- A – `compile-time, syntax`
- B – `compile-time, logic`
- C – `runtime, syntax`
- D – `runtime, logic`

Solution

What kind of error is in this program?

```
def Maximum(x,y):
    z=x
    if (x>y): z=y
    return z
```

- A – `compile-time, syntax`
- B – `compile-time, logic`
- C – `runtime, syntax`
- D – `runtime, logic` 

UNIVERSITY OF CAPE TOWN

IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

SCHOOL OF IT

Introduction to Programming

UNIVERSITY OF CAPE TOWN
IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

Testing and Debugging

Testing Approaches

CSC1

Hussein Suleman
Department of Computer Science

Testing Approaches

Hussein Suleman
Department of Computer Science
University of Cape Town

Errors: Recipe Analogy

Pancake Recipe:

30 ml olive oil

6 eggs

4 potatoes

1 clove minced garlic

Chop ptatos into small cubes. Vigourously eggs beat. Add chopped onion, garlic and oil to pan and fry till golden. Cut potatoes into thin slices. Pour egg into pan and cook till set.

Errors: Recipe Analogy

Pancake Recipe:

Logic error

30 ml olive oil
6 eggs
4 potatoes
1 clove minced garlic

Compile time errors

Chop **ptatos** into small cubes. **Vigourously eggs beat.** Add **chopped onion, garlic** and oil to **pan** and fry till golden. Cut potatoes into thin slices. Pour egg into pan and cook till set.

Run time error

Exercise

```
def Sorty(x,y,z):
    if x>y:
        if y>z:
            print(x y z)
        else:
            print(x,z,y)
            print(c)
    else:
        if y<z:
            print(z,y,x)
        else:
            print(y,x,z)
Sorty(1,2,3)
```

This program aims to sort three numbers into increasing order.

Can you find errors in this program? If so, list the type of error (syntax, runtime, etc.)

Exercise

```
def Sorty(x,y,z):
    if x>y:
        if y>z:
            print(x,y,z)
        else:
            print(x,z,y)
    else:
        if y<z:
            print(z,y,x)
        else:
            print(y,x,z)

Sorty(1,2,3)
```

After fixing the syntax errors, run this to uncover additional errors…

Supply different numbers to Sorty:
1,2,3
1,3,2
etc.

How many combinations are there?

Testing Methods

- Programs must be thoroughly tested for **all possible** input/output values to make sure the programs behave **correctly**.

Exhaustive testing

- Ideal testing strategy:
  - Run program using all possible inputs.
  - Compare actual outputs to expected outputs.

Exhaustive testing

- But how do we test for all values of integers?

```
def Threshold(val):
    if val>10000: val=10000
    return val
```

- This simple program asks the user for one integer.
  - How many possible input values are there in Python?

Random Testing

- A subset of values in the input domain is used for testing.
  - Important to ensure that values are distributed over input domain.
  - Can use random number generation.

UNIVERSITY OF CAPE TOWN

IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

SCHOOL OF IT

Introduction to Programming

UNIVERSITY OF CAPE TOWN
IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

Testing and Debugging

Equivalence Classes

CSC1

Hussein Suleman
Department of Computer Science

Equivalence Classes

Hussein Suleman
Department of Computer Science
University of Cape Town

Equivalence Classes & Boundary Values

- ***Equivalence classes***: Group input values into sets of values with similar expected behaviour and choose candidate values.
  - e.g.

    (100, -90, 1000)

    (40000, 100000)

```
def Threshold(val):
    if val>10000: val=10000
    return val
```

- ***Boundary value analysis***: Choose values at, and on either side of, the boundaries of the equivalence classes.
  - e.g. 9999, 10000, 10001

Equivalence Classes & Boundary Values

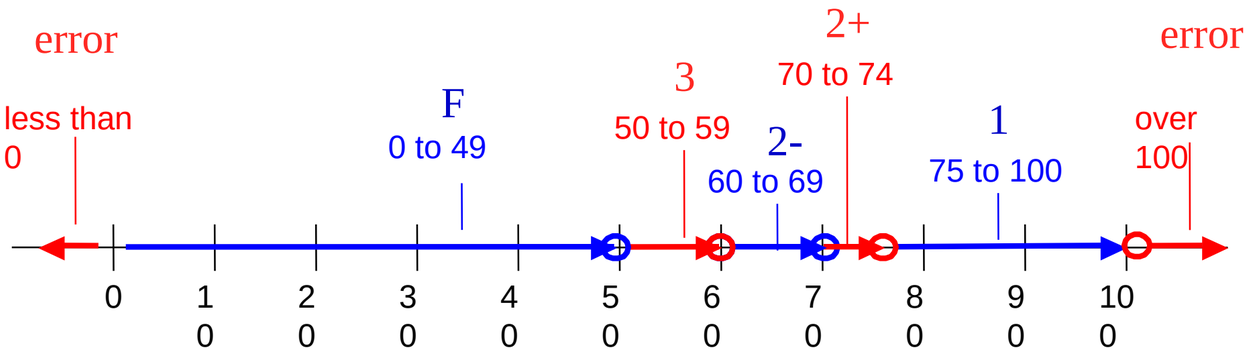
- Write a program to classify test scores into ranks:

  1, 2+, 2-, 3, F

Equivalence Classes & Boundary Values

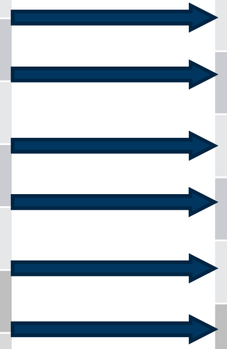
Example: Test score program

Equivalence classes:



Equivalence Classes & Boundary Values

Equivalence Class	*Sample Value*
Scores greater than 100	150
Scores between 75 and 100	95
Scores between 70 and 74	72
Scores between 60 and 69	65
Scores between 50 and 59	55
Scores between 0 and 49	30
Scores less than 0	-50



Boundary Value	*Just Above Boundary Value*	*Just Below Boundary Value*
100	101	99
75	76	74
70	71	69
60	61	59
50	51	49
0	1	-1

Exercise

```
def Sorty(x,y,z):
    if x>y:
        if y>z:
            print(x,y,z)
        else:
            print(x,z,y)
    else:
        if y<z:
            print(z,y,x)
        else:
            print(y,x,z)
```

What are the equivalence classes for this program?

What are the boundary values for this program?

Exercise

```
def Sorty(x,y,z):
    if x>y:
        if y>z:
            print(x,y,z)
        else:
            print(x,z,y)
    else:
        if y<z:
            print(z,y,x)
        else:
            print(y,x,z)
```

What are the equivalence classes for this program?

There are 6 classes. Examples are:
Sorty(3,2,1) #x>y>z
Sorty(3,1,2) #x>z>y
Sorty(2,1,3) #z>x>y

Sorty(2,3,1) #y>x>z
Sorty(1,3,2) #y>z>x
Sorty(1,2,3) #z>y>x

Boundaries: x=z, x=y, y=z, x=y=z

Exercise

```
def Sorty(x,y,z):
    if x>y:
        if y>z:
            print(x,y,z)
        else:
            print(x,z,y)
    else:
        if y<z:
            print(z,y,x)
        else:
            print(y,x,z)
```

Correct the Sorty function.

Call the function 6 times to test the different options:
Sorty (1,2,3)
Sorty (1,3,2)
Sorty (2,1,3)
Sorty (2,3,1)
Sorty (3,1,2)
Sorty (3,2,1)

UNIVERSITY OF CAPE TOWN

IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

SCHOOL OF IT

Introduction to Programming

UNIVERSITY OF CAPE TOWN
IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

Testing and Debugging

Path Testing and Statement Coverage

CSC1

Hussein Suleman
Department of Computer Science

Path Testing and Statement Coverage

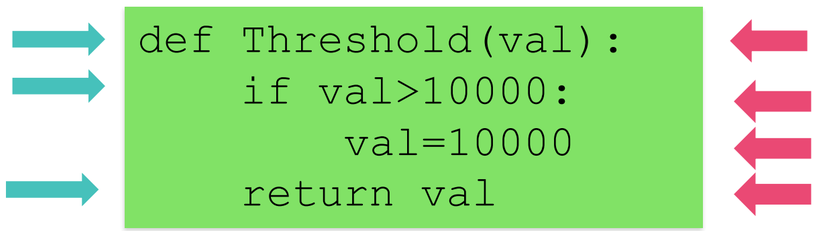
Hussein Suleman
Department of Computer Science
University of Cape Town

SCHOOL OF IT

UNIVERSITY OF CAPE TOWN
IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

Path Testing

- Create test cases to test **every path** of execution of the program at least **once**.



Path 1:

val = 35

Path 2:

val = 10001

Checkpoint

```
def Sorty(x,y,z):
    if x>y:
        if y>z:
            print(x,y,z)
        else:
            print(x,z,y)
    else:
        if y<z:
            print(z,y,x)
        else:
            print(y,x,z)
```

How many inputs would you need to do path testing for this program?

Statement Coverage

- What if we had:

```
if a<25:
   print ("Error in a")
else:
   print ("No error in a")
if b<25:
   print ("Error in b")
else:
   print ("No error in b")
```

- Rather than test all paths, test all statements at least once.
  - e.g., (a,b) = (10, 10), (50, 50)

Exercise

```
def riddle(n):
 if n<=0:
        return 0
 a=1
 b=1
 for i in range(n-2):
        a,b=a+b,a
 return a
```

What does this program do?

Equivalence classes:

Boundary values:

Statement coverage:

Path coverage:

Exercise

```
def riddle(n):
 if n<=0:
        return 0
 a=1
 b=1
 for i in range(n-2):
        a,b=a+b,a
 return a
```

Equivalence classes:

n <=0

0<n<3

n>=3

Boundary values:

-1,0,1,2,3,4

Statement coverage:

-5; 8

Path coverage:

-5;2;8

Glass and Black Boxes

If you can create your test cases based on only the problem specification, it is ***black box*** testing.

- If you have to examine the code, it is ***glass box testing***.

Which categories do these fall into?

- Exhaustive Testing
- Random Testing
- Equivalence classes/boundary values
- Path coverage
- Statement coverage

Poll

Which of these is the best approach to determine test values?

- A – `Exhaustive testing of all values`
- B – `Equivalence classes and boundary values`
- C – `Path testing`
- D – `Statement coverage`

Solution

Which of these is the best approach to determine test values?

- A – `Exhaustive testing of all values`
- B – `Equivalence classes and boundary values`
- C – `Path testing`
- D – `Statement coverage`

UNIVERSITY OF CAPE TOWN

IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

SCHOOL OF IT

Introduction to Programming

UNIVERSITY OF CAPE TOWN
IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

Testing and Debugging

Correcting Errors

CSC1

Hussein Suleman
Department of Computer Science

Correcting Errors

Hussein Suleman
Department of Computer Science
University of Cape Town

Finding Errors

- What if a test case fails? Now what?

- Find the error and remove it, using:
  - Tracing
  - Debugger

Tracing

- Insert temporary statements into code to output values during calculation.
- Very useful when there is no debugger!
- Example:

```
y = 4
x = y*y*2
z = x+5

print (z)
if z == 13:
    …
```

trace instruction

x	32
y	4
z	37
Z == 13	False

Screen Output:
37

Debugging

- **Debugging** is the process of finding **errors** or **bugs** in code.
  - A **debugger** is a tool for executing an application where the programmer can carefully control execution and inspect data.
  - Features include:
    - step through code one instruction at a time
    - viewing variables (“Stack Data” in Wing101)
    - insert and remove breakpoints to pause execution

Wing101 manual has info on debug

Exercise

- Track the execution of riddle:
  - Debugger
  - Tracing statements

Problem

- Write a program to convert a decimal number to binary.

Binary Codes

- Computers use presence/absence of voltage.
  - Possible values for digits: 0 and 1

Example:

- $10_2 = 1*2^1 + 0*2^0$
  $= 2_{10}$
- $1101 = 1*2^3 + 1*2^2 + 0*2^1 + 1*2^0$
  $=8+4+1$
  $=123$

Note: writing a number this way

10_2

means that the base is '2', i.e. it is a binary number.

An n-bit binary number can represent numbers from 0_{10} to $(2n-1)_{10}$

Decimal to Binary Conversion

Algorithm:

```
quot = number;
i = 0;
repeat until quot == 0
    quot = quot/2;
    digit_i = remainder;
    i++;
```

Example:

Convert 37_{10} to binary.

Calculation:

- 37/2 = 18 rem 1 least sig. digit
- 18/2 = 9 rem 0
- 9/2 = 4 rem 1
- 4/2 = 2 rem 0
- 2/2 = 1 rem 0
- 1/2 = 0 rem 1 most sig. digit

Result:

- $37_{10} = 100101_2$

Binary converter as a program

```
def convert_to_binary(decimal):
    result_string=""
    while decimal>0:
        remainder= decimal%2
        decimal=decimal//2
        result_string=str(remainder)+result_string
    return result_string
```

- *Use debugger to trace execution*

Exercise

- Determine test values for this function.

```
def convert_to_binary(decimal):
    result_string=""
    while decimal>0:
        remainder=decimal%2
        decimal=decimal//2
        result_string=str(remainder)+result_string
    return result_string
```

Equivalence classes:

Boundary values:

Statement coverage:

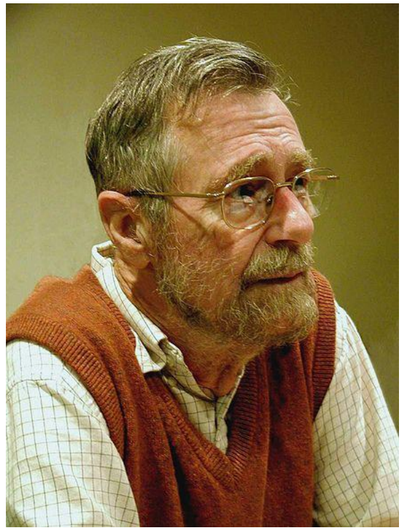
Path coverage:

Exercise

- Correct the Sorty function.

A quote to end the section

“Program testing can, at best, show the presence of errors, but never their absence.”



Edsger Dijkstra (1930-2002)
Dutch Computer Scientist

UNIVERSITY OF CAPE TOWN

IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

SCHOOL OF IT