

Answer the questions in the spaces provided on the question sheets. If you run out of room for an answer, continue on the back of the page. The exam has 4 questions with equal weight. Some questions have multiple parts. Point totals are given in the margin.

Name: \_\_\_\_\_ **Answer Key** \_\_\_\_\_

1. \_\_\_\_\_ **25**

2. \_\_\_\_\_ **25**

3. \_\_\_\_\_ **25**

4. \_\_\_\_\_ **25**

Total \_\_\_\_\_ **100**

No notes or books may be used on the exam. If you have any questions, please raise your hand and I will try to answer them.

5 1. (a) What are the five stages in the waterfall process model?

**Answer:**

- (requirements) analysis
- design
- implementation
- testing
- maintenance

5 (b) What is the difference between requirements and design?

**Answer:** Requirements determine what product should be built. Design determines how the product is built, ideally how a quality product is built.

5 (c) What is a software architecture?

**Answer:** A software architecture is a high-level or “macro” view of the system, which is usually simpler than the actual design.

5 (d) What are coupling and cohesion and how do they relate to the idea of quality software?

**Answer:** Coupling is a qualitative measurement of the dependence of one module upon other modules. Cohesion is a measurement of how well a module does a single, well-defined task or set of tasks. Quality software should exhibit low coupling and high cohesion.

(e) Define the following architectural terms:

1 i. adaptive

**Answer:** capable of adapting (interior agent)

1 ii. adaptable

**Answer:** capable of being adapted (exterior agent)

1 iii. extensible

**Answer:** capable of being used for purposes other than the intended

1 iv. extendible

**Answer:** capable of being built upon

1 v. scalable

**Answer:** capable of accommodating larger inputs

2. (a) Answer the following questions regarding the pipe and filter style.

3

i. List three advantages of the style.

**Answer:**

- simple to understand
- excellent reuse
- easily extended

3

ii. List three disadvantages of the style.

**Answer:**

- performance may be poor
- filters need to speak a common language
- ill-suited for interactive applications

2

iii. Give a basic invariant of the style.

**Answer:** filters are independent (i.e., have no shared data or knowledge); pipes are unidirectional (i.e., flow of data is unidirectional)

2

iv. Give a common specialization or example of the style.

**Answer:** pipelines provided by UNIX shells.

(b) Answer the questions regarding architectural styles:

5

i. Which styles would you choose if efficiency was most important? Why?

**Answer:** Shared data architectures and repositories are very efficient because there is typically only one copy of the data. Repositories also allow for parallelism. Object-oriented and event-based architectures can also be efficient. Pipe and filter and layered architectures are typically not very efficient. The former because a complete transformation of the input is required; the latter because crossing layers incurs a cost.

5

ii. Which styles would you choose if simplicity was most important? Why?

**Answer:** Pipe and filter architectures are probably the most simple since they describe a straightforward, step-by-step approach to solving a problem. Layered architectures can also be simple since they are modeled using abstraction. Event-based architectures are typically not very simple since control is distributed. Other architectures all have aspects that can make them relatively simple.

5

iii. Which styles would you choose if reusability was most important? Why?

**Answer:** Object-oriented architectures certainly have a high degree of reusability that is achieved through encapsulation and the grouping of data with operations. Pipe and filter architectures also exhibit high reusability since filters are independent. Other architectures are not as good at reuse, although all can be viewed as having some reuse to some degree.

3. (a) Indicate the location of
- `stdin`
- ,
- `stdout`
- , and
- `stderr`
- of each of the individual commands:

5

```
i. #!/bin/sh
{
  a | b | c > Z
} < X > Y 2>&1
```

**Answer:**

	a	b	c
<code>stdin</code>	X	pipe	pipe
<code>stdout</code>	pipe	pipe	Z
<code>stderr</code>	Y	Y	Y

10

```
ii. #!/bin/sh
{
  a > /dev/tty
  b < /dev/tty |
  c 2>&1 > /dev/null
} < X |
{
  d 1>&2
  (e | f) < X
} 2> Y
```

**Answer:**

	a	b	c	d	e	f
<code>stdin</code>	X	/dev/tty	pipe	pipe	X	pipe
<code>stdout</code>	/dev/tty	pipe	/dev/null	Y	pipe	/dev/tty
<code>stderr</code>	/dev/tty	/dev/tty	pipe	Y	Y	Y

(b) What is the output of the following shell scripts:

5

```
i. #!/bin/sh
cd /home
pwd
{ cd /tmp; pwd; }
pwd
( cd /var; pwd; )
pwd
```

**Answer:**

```
/home
/tmp
/tmp
/var
/tmp
```

5

```
ii. #!/bin/sh
false && echo X || echo Y
echo A || false && echo C
```

**Answer:**

```
Y
A
C
```

- 5 4. (a) Write a shell script, `revargs`, that writes on the standard output its arguments in reverse order. For example:

```
$ revargs one two three
three two one
```

Your script should be as robust as possible and deal with errors gracefully.

**Answer:**

```
#!/bin/sh

REV=""

for ARG in "$@"; do
    REV="$ARG $REV"
done

echo "$REV"
```

- 5 (b) Using the `revargs` script, write a shell script, `revcat`, that writes the contents of the files given as its arguments, but starting with the last file named and ending with the first file named. For example:

```
$ cat X Y Z
This is file X.
This is file Y.
This is file Z.
$ revcat X Y Z
This is file Z.
This is file Y.
This is file X.
```

Your script should be as robust as possible and deal with errors gracefully.

**Answer:**

```
#!/bin/sh

cat `revargs "$@"`
```

15

- (c) Write a script, `register`, that adds the contents of a file called `copyright` in the user's home directory to the beginning of each file listed as an argument. *Hint*: You will need to use a temporary file.

Your script should be as robust as possible and deal with errors gracefully.

**Answer:**

```
#!/bin/sh

if [ ! -f "$HOME/copyright" -o ! -r "$HOME/copyright" ]; then
    echo "$0: $HOME/copyright: not a readable file" 1>&2
    exit 1
fi

for FILE in "$@"; do
    if [ -f "$FILE" -a -r "$FILE" -a -w "$FILE" ]; then
        cat "$HOME/copyright" "$FILE" > /tmp/register &&
        mv /tmp/register "$FILE"
    else
        echo "$0: $FILE: not a readable, writable file" 1>&2
    elif
done
```