FOLD HERE

Surname: ....

Forename: ....

Prifysgol Abertawe
Swansea University

Faculty of Science and Engineering

Computer Science

CSC371 Advanced Object-Oriented Programming

May/June 2024

Student Number: ....

Seat Number: ....

Time Allowed: 2 hours

Do NOT open the paper until instructed to do so.

Exam Paper Information

Write your answers in the spaces provided.
If more space is required ask for a seperate answer book.
Answer all questions

Office Use Only	
Question	Mark

Special Instruction(s)

None

Specific Items

Dictionaries - Candidates may only refer to the English and Welsh language dictionaries available

Calculators - Candidates may NOT use a calculator

Open Book - This is NOT an Open Book Exam

Contact Details

Originator: ***Thomas Reitmaier***

Contact Number: ***01792606652 or contact the Exams Office***

NB. The content of this exam paper has been thoroughly checked, however if you think you have spotted an error, please raise your hand and report to an invigilator. Unless it has a significant impact on the paper the error will NOT be corrected in the exam venue. Errors not corrected in the exam should be noted in your answer booklet and you should continue to answer the examination paper to the best of your ability as it is written. It will be noted by the module coordinator and the Board of Examiners when it considers your results.

This page is intentionally left blank
(for double sided-printing)

Part 1. Multiple Choice Questions **[20 total marks]**

Q1. In line with departmental policy, multiple choice questions are not released. **[0 marks]**

Part 2. Written Answers **[30 total marks]**

Q2. Explain what the function `malloc` does and why it returns **`void`** `*`.

[2 marks]

Q3. Describe what a *smart pointer* does and why you would use one. Give an example of where smart pointers are used in standard library containers.

[3 marks]

Q4. Describe how *ad-hoc polymorphism* is an object-oriented programming principle and give **two** examples of how it is featured in C++.

[4 marks]

Q5. Consider the following C++ code:

```
template <typename T>                           1
void foo(T var) {                               2
    std::cout << "foo  = " << var << '\n';      3
}                                               4
```

Explain what types can be formally used by the template. Furthermore, explain any implications this has on line 3.

[2 marks]

Q6. Consider the following C++ code:

```
1   struct Functor {
2       inline int operator()(int x) const {
3           return x + 1;
4       }
5   };
6
7   int main() {
8       int i = 1;
9       // Your code goes here
10  }
```

Read the above valid C++ code and explain what a *functor* is and what this particular functor does. Write some code to go on line 9 to show how this functor can be called, passing in the integer parameter `i` provided for you on line 8.

[4 marks]

Q7. Explain the *Interface Segregation Principle* of programming and what its benefits are? Also give an example of a context of where it is used?
(You do not need to write code.)

[5 marks]

Q8. The next questions (**a**) – (**e**) all deal with the following, valid C++ code sample.

```
#include <iostream>
#include <string>

class Base {
protected:
  std::string name;

public:
  Base(std::string n) : name(n) {
     std::cout << "Base constructed\n";
  }

  virtual ~Base() { std::cout << "Base destructed\n"; }

  friend std::ostream &operator<<(std::ostream &os, const Base &pd);
};

class Derived : public Base {
public:
  Derived(std::string n) : Base(n) {
     std::cout << "Derived constructed\n";
  }

  virtual ~Derived() { std::cout << "Derived destructed\n"; }
};

std::ostream &operator<<(std::ostream &os, const Base &b) {
  return os << b.name << "\n";
}

int main() {
  Base *b = new Derived("Hello");
  std::cout << *b;

  delete b;
  return 0;
}
```

(**a**) What does the program output?

[3 marks]

(**b**) Explain what effect the keyword **`public`** has on line 18?

[2 marks]

(**c**) Name the idiom abbreviated as **RAII**.

[1 mark]

(**d**) To adhere to RAII principles, the destructors for `Base` (line 13) and `Derived` (line 24) have been declared as **`virtual`**. Explain why this is the case.

[2 marks]

(**e**) Describe the purpose of the **friend declaration** within C++. Also explain what error – if any – would occur (and why it would occur) if the **`friend`** keyword is omitted on line 15?

[2 marks]

End of Paper