



12 Pages

1. Please give the information requested in the boxes below.
2. Enter your name in the space provided to the right, fold over and seal down (using the label provided) to hide your name.
3. The answers must be legibly written. The Examiners can either ignore illegible script or have it typed and the expense of typing charged to the candidate.
4. Write your answers in ink.
5. Begin each answer on a fresh page. Do all rough work in the answer book and cross it through.
6. No part of this book is to be torn out; it must be handed in complete.
7. If more books than one are used, fasten them together using the tags available from the invigilator(s).
8. **Every book must be handed in. In no circumstances may answer books, used or unused, be taken out of the examination venue.**

1	Student Number	P147250
2	Title of Examination Paper	Programming
3	Module Code	MEX 8523
4	Date and Time of Examination	9am 18/3/18
5	Examination Venue	MB ~~103~~ 103
6	Seat Number	
7	Number of Answer Books handed in	1

List the numbers of the questions you have attempted	Office Use
Q1	6
Q2	5
~~Q3~~	~~5~~
Q4	4
Total	20

(Gweler yr ochr arall am y fersiwn Cymraeg)



Class Diagram



I think
this is
not your
asking
for...

PTO

Q1

Question No.

```
#include stdio <stdio>
#include ios <iosstream>
```

```
using namespace std;
```

```
class Customer ID
```

```
{
```

```
private:
```

```
int ID;
```

```
char ** Customer name;
```

```
int order ID;
```

```
int Stock level;
```

```
int Quantity;
```

```
int name;
```

```
public:
```

```
Customer ID() {
```

```
}
```

```
Customer ID (**name, int ID, int order ID, Pcoutlt)
```

```
{ Stock level = 0;
```

```
Customer = name [0];
```

```
order ID = ID;
```

```
Quantity = stock level;
```

```
}
```

```
void Stock levels()
```

```
{
```

```
if (stock level < Quantity)
```

```
{
```

```
cout << Quantity [stock level] << endl;
```

```
}
```

```
else
```

```
{
```

```
Stock level = 0;
```

```
}
```

* (here!)

```
& void Shipping (int ID, order
int order ID,
int name)
```

```
{
```

```
name
```

```
name = Customer name;
```

```
order ID = order ID;
```

```
}
```

```
};
```



Q2 int = Integer using in numbers
char = characters using a letter or string.

- a) Line: 1 - 3 - ostream is library & namespace is empty (also ~~the~~ ~~std~~ means standard.)
- Line: 8 - class called GameElement
- Line: 10 - 15 - private not for public to see & int
Integerate what it holds for function & char is character for function again.
- Line: 16 - 19 - public can see, open but empty.
- Line: 21 - uses functions that are in line 10 to 15.
- Line: 23 - 27 - are functions ^{used} ~~using~~ from line 10 to 15 to operate on what it's told to do.
- Line: 30 is same as Line 8 and 17.
- Line: 32 - void ~~means~~ empty but "DrawSprite" is a function working with void to create and argument.
- Line: 34 - "if" is a decision (making a choice) so it giving "pathCount" a decision to make if works.
- Line: 36 - 37 - cout is display to see it like "printf".
~~"printf" = print form~~ the "<<" are for to display what after it and endl is newline.
- Line: 39 and 41 else is same as "if" but opposite like "Yes and No" and line 41 is using line 23.
- Line: 46 - 50 - void ^{means} ~~is~~ empty but with another name with it and int used twice to what looks like "x and y" are giving coords that are used in line 12 and 13.
- Line: 50 is end of this class.

Q2

Question No.

b) ?



Examiner On

Q3

AGILE - ~~adapts~~ ~~adapts~~ adapts with customer Communications

Agile - good with small software programs

Agile - use stories

Agile - is not good with UML

~~UP is good with~~ approaches

UP is generally good ^ but Agile adapts within approaches

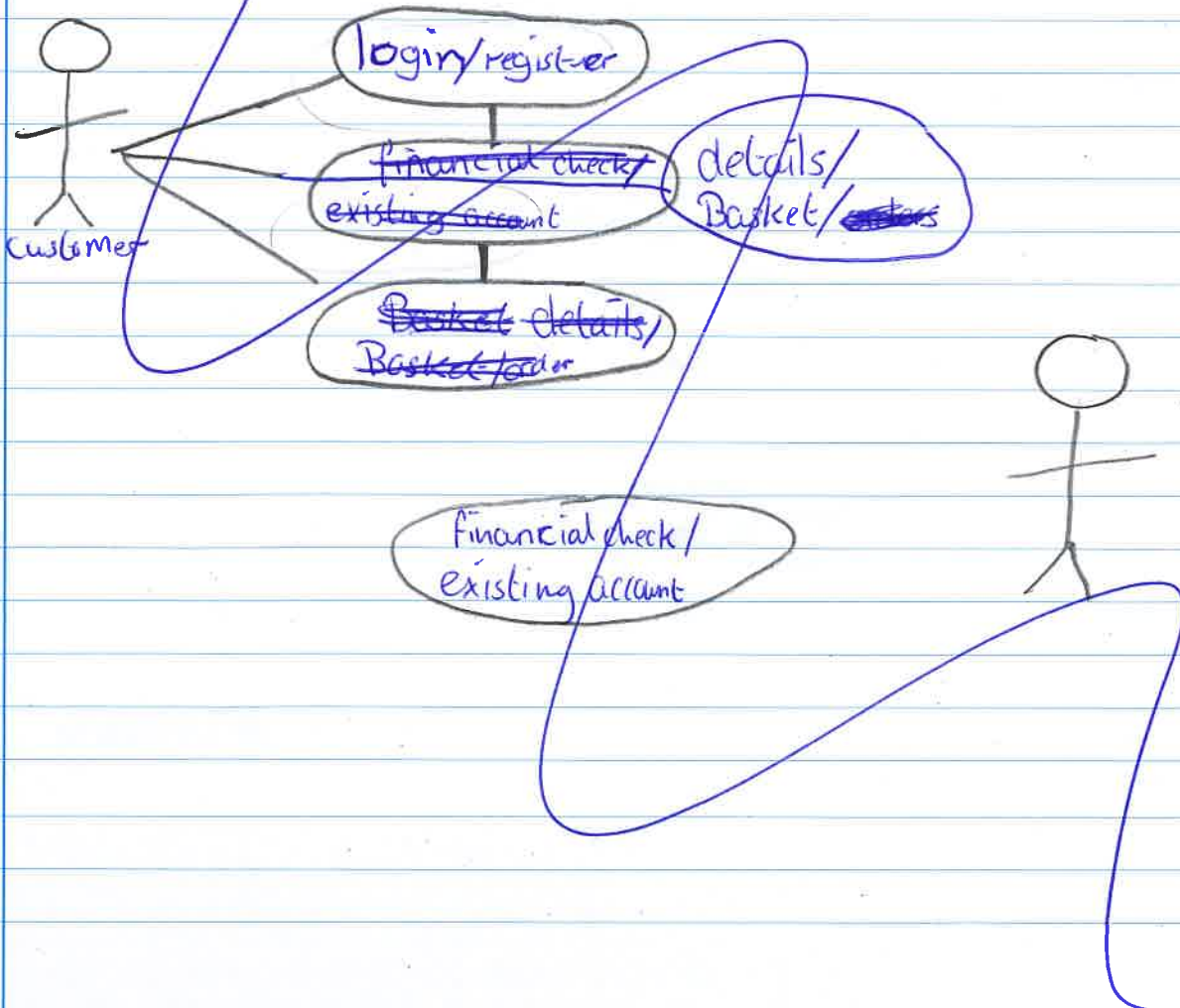
UP is better with long programs and and identify phases in programs but has issues with short programs but Agile is good with short but struggle with long programs

A process is ~~complex~~ complex

UP is closer to UML (unified model language)

Q4

Use Case diagram

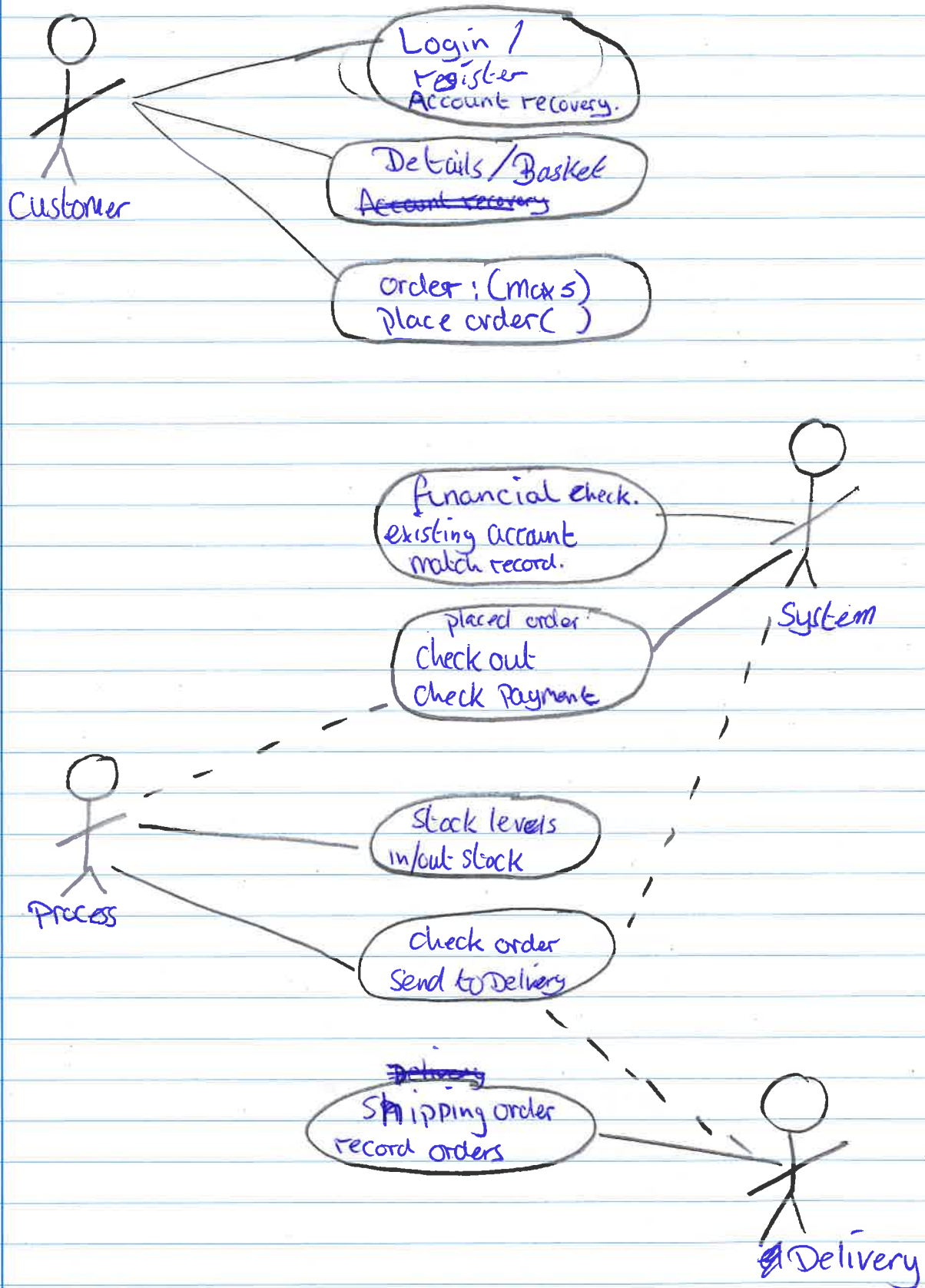


Q4

Question No.

Use Case diagram

a)

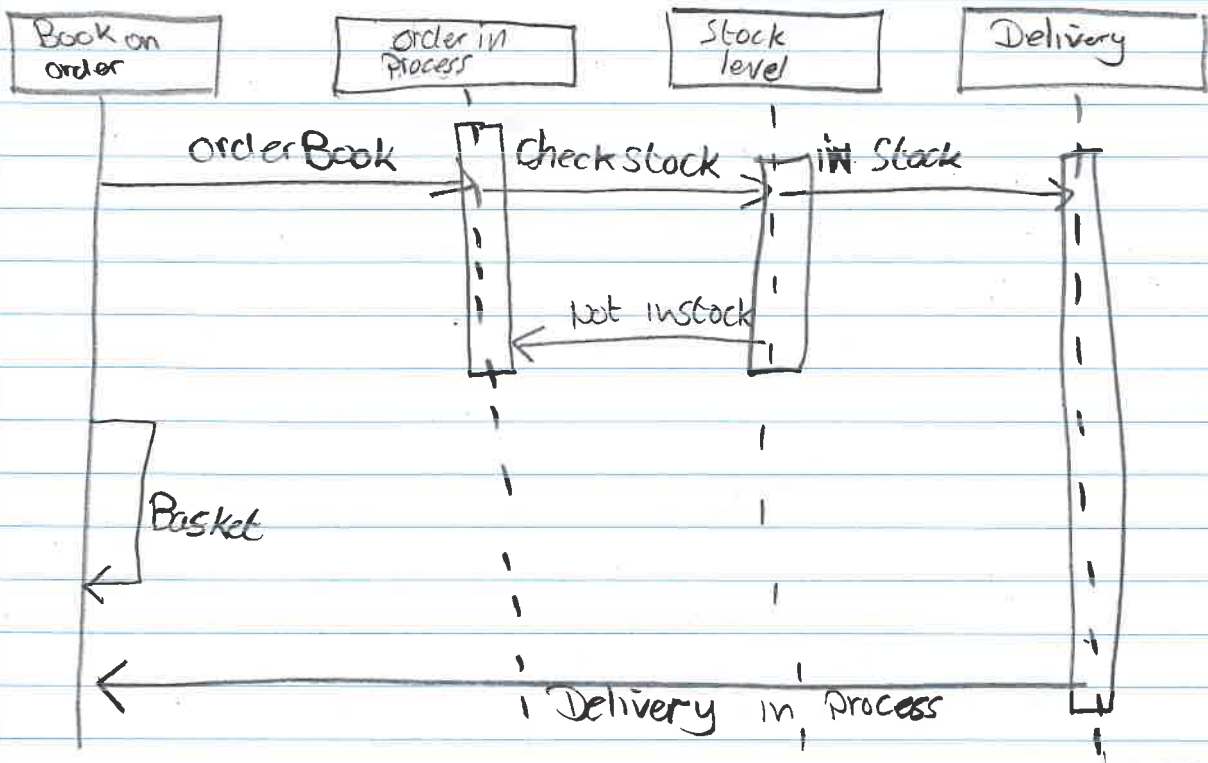


3

2



Q4
b)



4 ✓

Q4-9