Computing

GCSE

Computing

Exemplar Candidate Work

J275

Unit A453 Sample Material C2

Version 1

www.ocr.org.uk/gcsecomputing

OCR
Oxford Cambridge and RSA

Disclaimer on use of Sample Material

Confidentiality

These tasks are taken from legacy Controlled Assessment tasks, undertaken and submitted by candidates. Where possible, we have removed all identifying information from these assessments. Should any data remain, you are requested to treat this confidentially and inform OCR as soon as possible highlighting the data concerned.

Use of URS Sheets and Sample Material

These tasks have all been moderated as part of the relevant exam series in which they were submitted and the marks submitted have all been allowed to stand. However, schools should bear in mind that this only indicates that the **overall assessment** of the Controlled Assessment is within tolerance and not necessarily each individual mark band. There may be instances where the mark scheme has been applied too generously, or similarly too harshly. This would have been identified in the reports to the centre – but will not be evident from URS alone. The spirit of the release of these samples is to give teachers better understanding of what High, Medium and Low graded coursework would feel like as an entity, rather than exact definitions of requirements for mark bands independently.

The provision of high graded work should **not infer** that this is the only, or best way of writing up a Controlled Assessment Task. Candidates are encouraged to map their personal journey through the tasks. Writing frames, or 'guides' for documentation are against the spirit of the coursework and constitute malpractice.

Each set of materials released contains a High, Middle and Low grade band. This should allow teachers to gain good understanding of the general standard of work quality required for each mark band, and as a whole – especially when comparing each set side by side.

Teachers are encouraged to seek further support when they feel clarification is needed in applying the mark scheme. We would also recommend regular CPD in respect of Controlled Assessment delivery and marking.

Accuracy

All work has, where possible, remained unaltered from the original submission. There may well be grammatical errors and poor layout in diagrams. This is to allow better matching of mark band criteria, where specific bullet points refer to quality of Spelling, Punctuation and Grammar, and also ease of navigation etc. Any significant changes are clearly marked. Some data that is perceived sensitive may be blocked out in black.

GCSE Computing Controlled Assessment

Unit A453 Programming project

Unit Recording Sheet

Please read the instructions printed on the other side of this form. **One** of these Unit Recording Sheets, suitably completed, should be attached to the assessed work of **each** candidate.

Unit	**A453**		**Year**	
Centre Name			**Centre Number**	
Candidate Name			**Candidate Number**	

	Guidance			Teacher Comment	Mark
Use of programming techniques	There is an attempt to solve parts of the tasks using few of the techniques identified. 0 = no response or responses not worthy of credit **[0 - 2]**	There is an attempt at most parts of the tasks using several techniques. **[3 - 4]**	There is an attempt to solve all of the tasks using most of the techniques listed. **[5 - 6]**	The first 2 tasks have been solved, the third one is incomplete. most of the techniques have been used	4 **Max 6**
Efficient use of programming techniques	The techniques used produce partially working solutions to a small part of the problem. 0 = no response or responses not worthy of credit **[0 - 4]**	The techniques are used appropriately giving working solutions to most of the parts of the problem. Some sections of the solution are inefficiently coded. **[5 - 8]**	The techniques are used appropriately in all cases giving an efficient, working solution for all parts of the problem. **[9 - 12]**	All of the techniques used are appropriate for the tasks. and provide a working solution for the first two tasks, and a partial solution for the third task. While the code is efficient in places for the solutions, The techniques used lack description	7 **Max 12**

Design	There are comments on what the task involves and a limited outline describing the intended approach to some parts of the problem. There are brief comments on how this might be tested but with no mention of success criteria. 0 = no response or responses not worthy of credit **[0 - 3]**	There is a brief analysis of the tasks indicating what is required for each of the tasks. There is a set of basic algorithms outlining a solution to most parts of the problem. There is some discussion of how this is tested and how this compares to the identified outcomes in the tasks. There is discussion of the variables to be used and some general discussion of validation. **[4 - 6]**	There is a detailed analysis of what is required for these tasks justifying their approach to the solution. There will be a full set of detailed algorithms representing a solution to each part of the problem. There is detailed discussion of testing and success criteria. The variables and structures are identified together with any validation required **[7 - 9]**	Some analysis has taken place for each task. Psuedocode has been used to demonstrate how the program will function, although it lacks description in places. success criteria is included in testing tables. validation is identified in testing screens	6 **Max 9**
Development	There is some evidence to show a solution to part of the problem with some evidence to show that it works. Code is presented with little or no annotation, the variable names not reflecting their purpose and with little organisation or structure. 0 = no response or responses not worthy of credit **[0 - 3]**	There is evidence to show how the solutions were developed. There is some evidence of testing during development showing that many parts of the solution work. The code is organised with sensible variable names and with some annotation indicating what sections of the code does. **[4 – 6]**	There is detailed evidence showing development of the solution with evidence of systematic testing during development to show that all parts work as required. The code is well organised with meaningful variable names and detailed annotation indicating the function of each section. **[7- 9]**	Testing shows development of programs, and also working ability of those programs. code uses sensible variable names, there is some annotation on the code, but it does lack some description in places.	5 **Max 9**

Testing and evaluation	There is evidence to show that the system has been tested for function but the test plan is limited in scope with little structure. There is limited evidence to show how the result matches the original criteria. The evidence of written communication is limited with little or no use of specialist terms. Errors in spelling, punctuation and grammar may be intrusive. Information may be ambiguous or disorganised. 0 = no response or responses not worthy of credit **[0 - 3]**	There is a test plan covering many parts of the problem with some suggested test data. There is evidence that the system has been tested using this data. There is some evidence to show how the results of testing have been compared to the original criteria. There is a brief discussion of how successful or otherwise the solutions are. The quality of written communication is good using some specialist terms. There are few errors in spelling, grammar and punctuation. Information for the most part is presented in a structured format. **[4 - 6]**	The test plan covers all major success criteria for the original problem with evidence to show how each of these criteria have been met, or if they have not been met, how the issue might be resolved. There is a full evaluation of the final solution against the success criteria. A high level of written communication is obvious throughout the task and specialist terms/ technology with accurate use of spelling will have been used. Grammar and punctuation are used correctly and information is presented in a coherent and structured format. **[7 - 9]**	Testing has been undertaken for all solutions, but without thorough testing for all each solution it can be hard to correlate tests to outcomes. some of the components have been evaluated against the success criteria, Report is easy to follow and development is shown throughout.	6 **Max 9**
				Total/45	**28**

Guidance on Completion of this Form

1 **One** sheet should be used for each candidate.
2 Please ensure that the appropriate boxes at the top of the form are completed.
3 Using the guidance identify the most appropriate mark range for the work and enter the mark awarded for each element in the mark column.
4 Add appropriate comments to assist the moderator in the 'Teacher Comment' column.
5 Add the marks for the strands together to give a total out of 45. Enter this total in the relevant box.

A453

Task 1 – Binary / Decimal converter

Design, code, test and evaluate a program (or programs) that will convert between binary and decimal. For the decimal to binary converter the program should accept a positive value and output the binary equivalent. The system need only be tested for values up to 255.

What is binary?
Binary consists of two numbers which are 1 and 0. These are the numbers the computer understands.

For this task I will be using python to create the program for this. Python is a high level programming language.

The outcome of this task should be that my program should be able to covert binary to decimal and decimal to binary. In the binary to decimal it has to take an 8 bit binary an example of this would be (00010001) and then has to change it to a decimal (17). In the decimal to binary the user has to input a number between 1-255. An example of this would be a decimal (13) and this would then be converted into a binary (00001101)

Pseudo code:
Print 'binary to decimal converter'
Bit 8 (input 1 or 0)
While bit != 1 or 0
Ask (input the correct binary number)
Repeat bits until user has inputted the 8 bits
One=Bit 1 *1
Two=Bit 2 *1*2
Three=Bit 3 *1*2*2
Four=Bit 4 *1*2*2*2
Five=Bit 5 *1*2*2*2*2
Six=Bit 6 *1*2*2*2*2*2
Seven=Bit 7 *1*2*2*2*2*2*2
Eight=Bit 8 *1*2*2*2*2*2*2*2
Print ('the decimal value is',one+two+three+four+five+six+seven+eight)

Test no.	description	Data	Expected outcome
1	This part of the program tells the user the input the answer and then it'll take them to the chosen converter and if it is the binary converter it will ask them for an input of an 8 bit and if the user inputs more or less then it asks them for another input again	10001111	true
2	This is meant to print out the binary and it also checks if the numbers are 0 and 1	10012001	False it has an invalid syntax
3			true
4	This is the whole program(for binary to decimal) this is where it is to check if the numbers are 0s and 1s and also it will add it up and print out the decimal number.	10011001 → 153	The outcome is now true

Test no.1

```
  print('binary/decimal converter')
swer = str(input('would you like to convert binary to decimal or decimal to binary'))#t
 answer == 'binary to decimal':#an if statement would check if th econdition is true or
  binary = int(input('choose an 8 bit binary number'))
  x = binary
ile len(str(x)) >8:#this will check the length of the binary that is inputed by the use
  print ('it is too long')#if the binary is too long the user will be asked to input an
  binary1 = int(input('please enter a 8 bit binary again'))#this will be repeated uniti
int('make sure the binatry is an 8 bit meaning it consists of 8 numbers')
```

This will tell the user what the program is.

Test no.2

```
lef coversion():
    print('binary/decimal converter')
nswer = str(input('would you like to convert binary to decimal or decimal to binary')
.f answer == 'binary to decimal':#an if statement would check if th econdition is true
    binary = int(input('choose an 8 bit binary number'))
    x = binary
hile len(str(x)) >8:#this will check the length of the binary that is inputed by the
    print ('it is too long')#if the binary is too long the user will be asked to input
    binary1 = int(input('please enter a 8 bit binary again'))#this will be repeated un
rint('make sure the binatry is an 8 bit meaning it consists of 8 numbers')

hile bit8!=0 and bit!=1:
        print('invalid number please enter another binary that consists of 1s and 0s')
        bit8= int(input(" the digit is ..."))
hile bit7!=0 and bit!=1:
        print('invalid number please enter another binary that consists of 1s and 0s')
        bit7= int(input(" the digit is ..."))
hile bit6!=0 and bit!=1:
        print('invalid number please enter another binary that consists of 1s and 0s')
        bit6= int(input(" the digit is ..."))
hile bit5!=0 and bit!=1:
        print('invalid number please enter another binary that consists of 1s and 0s')
        bit5= int(input(" the digit is ..."))
hile bit4!=0 and bit!=1:
        print('invalid number please enter another binary that consists of 1s and 0s')
        bit4= int(input(" the digit is ..."))
hile bit3!=0 and bit!=1:
        print('invalid number please enter another binary that consists of 1s and 0s')
        bit3= int(input(" the digit is ..."))
hile bit2!=0 and bit!=1:
        print('invalid number please enter another binary that consists of 1s and 0s')
        bit2= int(input(" the digit is ..."))
hile bit1!=0 and bit!=1:
        print('invalid number please enter another binary that consists of 1s and 0s')
        bit1= int(input(" the digit is ..."))
```

I have removed this part of the program this is because I changed the way the program was set out.

I think by removing this it made it easier for me to make the rest of the program this is because I would have to break the 8 numbers up and then put them in order, but by using the separate bits it makes the calculations easier.

Test no.3

```
while bit8!=0 and bit!=1:
    print('invalid number please enter another binary that consists of 1s and 0s')
    bit8= int(input(" the digit is ..."))
while bit7!=0 and bit!=1:
    print('invalid number please enter another binary that consists of 1s and 0s')
    bit7= int(input(" the digit is ..."))
while bit6!=0 and bit!=1:
    print('invalid number please enter another binary that consists of 1s and 0s')
    bit6= int(input(" the digit is ..."))
while bit5!=0 and bit!=1:
    print('invalid number please enter another binary that consists of 1s and 0s')
    bit5= int(input(" the digit is ..."))
while bit4!=0 and bit!=1:
    print('invalid number please enter another binary that consists of 1s and 0s')
    bit4= int(input(" the digit is ..."))
while bit3!=0 and bit!=1:
    print('invalid number please enter another binary that consists of 1s and 0s')
    bit3= int(input(" the digit is ..."))
while bit2!=0 and bit!=1:
    print('invalid number please enter another binary that consists of 1s and 0s')
    bit2= int(input(" the digit is ..."))
while bit1!=0 and bit!=1:
    print('invalid number please enter another binary that consists of 1s and 0s')
    bit1= int(input(" the digit is ..."))

    bit1=1*1
    bit2=1*2
    bit3=1*2*2
    bit4=1*2*2*2
    bit5=1*2*2*2*2
    bit6=1*2*2*2*2*2
    bit7=1*2*2*2*2*2*2
    bit8=1*2*2*2*2*2*2*2
```

Each bit saves the binary number that is inputted and then times it by whatever bit it is. If it is a 0 it will equal to zero if it is 1 it equals to a decimal number

Test no.4

```
def coversion():

    print('binary/decimal converter')
answer = str(input('would you like to convert binary to decimal or decimal to binary'))#this w
if answer == 'binary to decimal':#an if statement would check if th econdition is true or fals
    print('enter each value one by one')
    bit8=int(input('the first digit'))
    while bit8!=0 and bit8!=1:
        print('invalid number please enter another binary that consists of 1s and 0s')#if the
        bit8= int(input(" the digit is ..."))
    bit7=int(input('the second digit'))
    while bit7!=0 and bit7!=1:
        print('invalid number please enter another binary that consists of 1s and 0s')
        bit7= int(input(" the digit is ..."))
    bit6=int(input('the third digit'))
    while bit6!=0 and bit6!=1:
        print('invalid number please enter another binary that consists of 1s and 0s')
        bit6= int(input(" the digit is ..."))
    bit5=int(input('the fourth digit'))
    while bit5!=0 and bit5!=1:
        print('invalid number please enter another binary that consists of 1s and 0s')
        bit5= int(input(" the digit is ..."))
    bit4=int(input('the fifth digit'))
    while bit4!=0 and bit4!=1:
        print('invalid number please enter another binary that consists of 1s and 0s')
        bit4= int(input(" the digit is ..."))
    bit3=int(input('the sixth digit'))
    while bit3!=0 and bit3!=1:
        print('invalid number please enter another binary that consists of 1s and 0s')
        bit3= int(input(" the digit is ..."))
    bit2=int(input('the seventh digit'))
    while bit2!=0 and bit2!=1:
        print('invalid number please enter another binary that consists of 1s and 0s')
        bit2= int(input(" the digit is ..."))
    bit1=int(input('the eighth digit'))
    while bit1!=0 and bit1!=1:
        print('invalid number please enter another binary that consists of 1s and 0s')
        bit1= int(input(" the digit is ..."))

    one=bit1*1
    two=bit2*1*2
    three=bit3*1*2*2
    four=bit4*1*2*2*2
    five=bit5*1*2*2*2*2
    six=bit6*1*2*2*2*2*2
    seven=bit7*1*2*2*2*2*2*2
    eight=bit8*1*2*2*2*2*2*2*2
    print('the decimal for your binary number is',one+two+three+four+five+six+seven+eight)
```

Bit 1 equals to 1. Bit 2 equals to 2. Bit 3 equals to 4. Bit 4 equals to 8. Bit 5 equals to 16. Bit 6 equals to 32. Bit 7 equals to 64. Bit 8 equals to 128.

Prints out the decimal this is done by adding all of the eight bits to get the final answer.

```
>>>
would you like to convert binary to decimal or decimal to binarybinary to decima
l
enter each value one by one
the first digit1
the second digit0
the third digit0
the fourth digit1
the fifth digit1
the sixth digit0
the seventh digit0
the eighth digit1
the decimal for your binary number is 153
>>>
```

Full Program:

```
def binary(n):

answer = str(input('would you like to convert binary to decimal or decimal to binary'))#this will ask the user to input the choice of conversion they want
if answer == 'binary to decimal':#an if statement would check if the condition is true or false
    print('enter each value one by one')
    bit8=int(input('the first digit'))
    while bit8!=0 and bit8!=1:
        print('invalid number please enter another binary that consists of 1s and 0s')#if the numbers are invalid then they are asked to input another set of 1s and 0s
        bit8= int(input(" the digit is ..."))
    bit7=int(input('the second digit'))
    while bit7!=0 and bit7!=1:
        print('invalid number please enter another binary that consists of 1s and 0s') #if the numbers are invalid then they are asked to input another set of 1s and 0s
        bit7= int(input(" the digit is ..."))
    bit6=int(input('the third digit'))
    while bit6!=0 and bit6!=1:
        print('invalid number please enter another binary that consists of 1s and 0s') #if the numbers are invalid then they are asked to input another set of 1s and 0s
        bit6= int(input(" the digit is ..."))
    bit5=int(input('the fourth digit'))
    while bit5!=0 and bit5!=1:
        print('invalid number please enter another binary that consists of 1s and 0s') #if the numbers are invalid then they are asked to input another set of 1s and 0s
        bit5= int(input(" the digit is ..."))
    bit4=int(input('the fifth digit'))
    while bit4!=0 and bit4!=1:
        print('invalid number please enter another binary that consists of 1s and 0s') #if the numbers are invalid then they are asked to input another set of 1s and 0s
        bit4= int(input(" the digit is ..."))
    bit3=int(input('the sixth digit'))
    while bit3!=0 and bit3!=1:
        print('invalid number please enter another binary that consists of 1s and 0s')
        bit3= int(input(" the digit is ..."))
    bit2=int(input('the seventh digit'))
```

```
    while bit2!=0 and bit2!=1:
        print('invalid number please enter another binary that consists of 1s and 0s') #if the numbers are invalid then they are asked to input another set of 1s and 0s
        bit2= int(input(" the digit is ..."))
    bit1=int(input('the eighth digit'))
    while bit1!=0 and bit1!=1:
        print('invalid number please enter another binary that consists of 1s and 0s') #if the numbers are invalid then they are asked to input another set of 1s and 0s
        bit1= int(input(" the digit is ..."))
    one=bit1*1 #multiplies the bit number and assigns it to the variable
    two=bit2*1*2#multiplies the bit number and assigns it to the variable
    three=bit3*1*2*2#multiplies the bit number and assigns it to the variable
    four=bit4*1*2*2*2#multiplies the bit number and assigns it to the variable
    five=bit5*1*2*2*2*2#multiplies the bit number and assigns it to the variable
    six=bit6*1*2*2*2*2*2#multiplies the bit number and assigns it to the variable
    seven=bit7*1*2*2*2*2*2*2#multiplies the bit number and assigns it to the variable
    eight=bit8*1*2*2*2*2*2*2*2#multiplies the bit number and assigns it to the variable
    print('the decimal for your binary number is',one+two+three+four+five+six+seven+eight) #outputs the total numbers
```

I have faced many difficulties in this task as I had to make many changes in this. These changes have not affected my program but have made them better and this has helped me improve it.

First the program will print out what is the task that is being done and then asks the user to input the desired conversion wanted and as this is the binary to decimal converter I will be explaining what is happening. The user is then asked to input the binary values one by one and as the numbers are inputted they are saved as one of the eight bits. Once the user has put the number wanted it will check if it is a 1 or a 0 and if the user does not input a 1 or a 0 they are then asked to input the number again until they get the correct number. Once the number has been inputted it is saved under the bit and then each bit is then times by 2 as each bit gets to the next bit the amount of times 2 increases by one and then this work out the decimal part of the solution. As each bit has the binary saved it will then work out what bits have a 1 and a 0. If the bit has a 0 it will equal to 0 but if the bit equals to 1 it will work out the number. At the end of the program it will print out the decimal.

Task 2 – Adding binary numbers

Design, code, test and evaluate a program that will accept two binary values (up to 8 binary digits) and output their total in binary. The output should not contain any leading zeros.

Pseudo code:
Print "adding binary"
Ask user to "input the binary number one by one"
While != 1 or 0
Ask user to input again
Once the user has inputted
The input number should be multiplied and divided 2
Print "the added binary"

Test no.	description	Data	Expected outcome
1	The program should ask the user to input the binary desired binary numbers.	1000 1000	true
2	It should be able to add the user's numbers together to make the binary.	00010000	false
3			True

Test no.1

```
outputm = 8

while output >= 1 :
    binary=int(input('enter binary'))
    number = number + (output*binary)
    output = output / 2

print('+')

while outputm >= 1 :
    binarym=int(input('enter binary'))
    numberm = numberm + (outputm*binarym)
    outputm = outputm / 2

    amount = output + outputm
```

This is to show that the binary will be added together

This is where the user is asked to input the number they want and this has been successful as it asks for an input the correct amount of times

Test no.2

```
amount= output + outputB
m1 = int (amount / 2)
b1 = int (amount % 2)
m2 = int (m1 // 2)
b2 = int (m1 % 2)
m3 = int (m2 // 2)
b3 = int (m2 % 2)
m4 = int (m3 // 2)
b4 = int (m3 % 2)
m5 = int (m4 // 2)
b5 = int (m4 % 2)
m6 = int (m5 // 2)
b6 = int (m5 % 2)
m7 = int (m6 // 2)
b7 = int (m6 % 2)
m8 = int (m7 // 2)
b8 = int (m7 % 2)
print('')
print('=')
print('')
print('{0}{1}{2}{3}{4}{5}{6}{7}'.format(b8,b7,b6,b5,b4,b3,b2,b1))
```

The outcome that was expected did not come out due to the 'total' this is because this variable was not defined and was meant to be amount instead to work.

Test no.3

```
    amount = output + outputm
m1 = int (amount / 2)
b1 = int (amount % 2)
```

Changed variable and now the program works and prints out the binary added

```
>>> ================================ RESTART ================================
>>>
enter binary1
enter binary0
enter binary0
enter binary0
+
enter binary1
enter binary0
enter binary0
enter binary0

=

00010000
```

This is the outcome of the program. This adds the binary numbers to make an 8 bit binary.

Full Program:

```
def binaryaddition():

    print('binary addition')#displays program name
    number = 0 #assigns variable
    output = 8 #assigns variable
    numberM = 0 #assigns variable
    outputB = 8 #assigns variable
    while output >=1 : #asks for 8 bits
        bit = int(input('enter binary')) #user enters binary number
        number = number + (output * bit)
        output = output / 2 #divides output
        print('')
        print('+')
        print('')
        while outputB >=1 : #asks for 8 bits
            bit2 = int(input('enter binary ')) #user enters binary number
            numberM = numberM + (outputB * bit2)
            outputB = outputB / 2
        amount= output + outputB
        m1 = int (amount / 2) #divides number assigns it to variable
        b1 = int (amount % 2) #divides number assigns it to variable
        m2 = int (m1 // 2) #divides number assigns it to variable
        b2 = int (m1 % 2) #divides number assigns it to variable
        m3 = int (m2 // 2) #divides number assigns it to variable
        b3 = int (m2 % 2) #divides number assigns it to variable
        m4 = int (m3 // 2) #divides number assigns it to variable
        b4 = int (m3 % 2) #divides number assigns it to variable
        m5 = int (m4 // 2) #divides number assigns it to variable
        b5 = int (m4 % 2) #divides number assigns it to variable
        m6 = int (m5 // 2) #divides number assigns it to variable
        b6 = int (m5 % 2) #divides number assigns it to variable
        m7 = int (m6 // 2) #divides number assigns it to variable
        b7 = int (m6 % 2) #divides number assigns it to variable
        m8 = int (m7 // 2) #divides number assigns it to variable
        b8 = int (m7 % 2) #divides number assigns it to variable
        print('')
        print('=')
        print('')
        print('{0}{1}{2}{3}{4}{5}{6}{7}'.format(b8,b7,b6,b5,b4,b3,b2,b1)) #prints out binary string
```

In task two I found that this was the hardest challenge to figure out as I had to face many difficulties to gain the results. The changes made have helped to improve the program.

First the program will ask the user to input the binary number they want added. In the program they are asked to input the numbers one at a time.

Task 3 – Binary logic

A food vending machine accepts 10p, 20p, 50p and £1 coins.
One or more coins are inserted and the current credit is calculated and displayed.
A product is selected from those available. The system checks to see if there is enough credit to
Purchase the product chosen.
If there is not enough credit the system displays an error message.
If there is enough credit it dispenses the product, updates the credit available and displays the remaining credit.
Further selections can be made if there is enough credit.
The vending machine simulation should have five products and prices.
Design, code, test and evaluate a program for this simulation.

For this task I am also going to be using python program it. For this task I am going to show the user what is available and the prices for the food. Then it asks what they want and then the price will also show up and then they have to input the amount using £1.00, 50p, 20p and 10p.the program then needs to add up the total amount and tell the user how much needs to be inputted.

Pseudo code:
Print "vending machine"
Print "prices and the food available in the vending machine"
After the user has inputted the desired item
Print "it costs ……"
While food value is …..
Print "input the correct amount"
If amount != certain amount
Print "please input the correct amount"
Then output what the user had asked from the vending machine
Example: print " you have got crisps and water"

Shows the user what can be inputted into it and what is available to buy in the program.

```
print('This vending machine only accepts 10p, 20p, 50p and £1 coins.')#as this p
print('this vending machine has crisp for 70p')#these items have the prices for
print('choclate is for 60p')
print('a drink is for 80p')
print('a snadwhich is £1.80')
food = str(input('what would you like to buy'))#this will ask the user what they
```

This will print out the options available and it also shows the prices. The next part then asks them what they want.

```
def vending():

    print('This vending machine only accepts 10p, 20p, 50p and 100p coins.')#as
print('this vending machine has crisp for 50p, choclate is for 30p, a drink is f
food = str(input('what would you like to buy'))#this will ask the user what they
if food == 'crisp':
    crisp = int(input('insert 50p'))#if the user chooses a certian type of food
value = [10,20,50,100]
if value == 50:
        print('this is the right amount')
while value != 50:
  print('this is wrong')
crisp = int(input('insert the right amount again'))
if food == 'chocolate':
    chocolate = int(input('insert 30p'))
value = [10,20,50,100]
if value == 30:
        print('this is the right amount')
while value != 30:
  print('this is wrong')
chocolate = int(input('insert the right amount again'))
if drink == 'chocolate':
    drink = int(input('insert 60p'))
value = [10,20,50,100]
if value == 60:
        print('this is the right amount')
while value != 60:
  print('this is wrong')
drink = int(input('insert the right amount again'))
if food == 'sandwich':
    sandwich = int(input('insert 50p'))
value = [10,20,50,100]
if value == 100:
        print('this is the right amount')
while value != 100:
  print('this is wrong')
sandwich = int(input('insert the right amount again'))
```

Tells the user what options are available and how much they cost and what the vending machine accepts.

Whatever food that is chosen will show the price that needs to be inputted it will then take the value inputted by the user and check if it is correct if not then the user is asked to input the correct value.

The while loop will keep it going until the user has inputted the correct amount of change.

A list is put in to this because it there to make sure the user inputs these amounts rather than just any random number.

Full Program

```
def vending():

    count = 0 #assigns variable
    totalcredit = 0 #assigns variable
    coinnum = int(input ('how many coins would you like to enter: ')) #user inputs coins
    while count in range (coinnum) :
        coin = float (input ('enter coin: £ '))
        totalcredit = totalcredit + coin
        count = count + 1
        print(' you have £(0) ' .format(round(totalcredit, 2)))
        print('')
        print(' what do you want to buy')
        print('')
        print('1.kit-kat')
        print('2.water')
        print('3.crisp')
        print('4.sandwich')
        print('')
        finalcredit = totalcredit
        round (finalcredit, 2)
        item = int (input('enter the item you want'))
        while item <1 or item >4: #only lets user choose the items
            print('this item cannot be bought')
            item = int (input('enter the item you want'))
        if item
```

In this program I had to change some of it up. This is because it wouldn't take the input at first and then as I changed it. It helped to improve it. First the program asks the user what they want from the vending machine and it tells them how much the machine accepts. As the user inputs the food they want, it will ask them to input the correct amount of money and if they do not it will ask them to input it until they get the correct amount. Unfortunately I could not complete the full program in the time allocated, with more time I could of finished it and got some more tests.

The small print

We'd like to know your view on the resources we produce. By clicking on the 'Like' or 'Dislike' button you can help us to ensure that our resources work for you. When the email template pops up please add additional comments if you wish and then just click 'Send'. Thank you.

If you do not currently offer this OCR qualification but would like to do so, please complete the Expression of Interest Form which can be found here: www.ocr.org.uk/expression-of-interest

OCR Resources: *the small print*

OCR's resources are provided to support the teaching of OCR specifications, but in no way constitute an endorsed teaching method that is required by the Board and the decision to use them lies with the individual teacher. Whilst every effort is made to ensure the accuracy of the content, OCR cannot be held responsible for any errors or omissions within these resources. We update our resources on a regular basis, so please check the OCR website to ensure you have the most up to date version.

OCR acknowledges the use of the following content:

Square down and Square up: alexwhite/Shutterstock.com

Please get in touch if you want to discuss the accessibility of resources we offer to support delivery of our qualifications: resources.feedback@ocr.org.uk

We will inform centres about any changes to the specification. We will also publish changes on our website. The latest version of our specification will always be the one on our website (www.ocr.org.uk) and this may differ from printed versions.

ocr.org.uk/alevelreform
OCR customer contact centre

General qualifications

Telephone 01223 553998
Facsimile 01223 552627
Email general.qualifications@ocr.org.uk

OCR is part of Cambridge Assessment, a department of the University of Cambridge. *For staff training purposes and as part of our quality assurance programme your call may be recorded or monitored.* © **OCR 2015** Oxford Cambridge and RSA Examinations is a Company Limited by Guarantee. Registered in England.
Registered office 1 Hills Road, Cambridge CB1 2EU. Registered company number 3484466. OCR is an exempt charity.

A DIVISION OF CAMBRIDGE ASSESSMENT

LLOYD'S REGISTER · LRQA ISO 9001 UKAS MANAGEMENT SYSTEMS 001