

## CSE1301 Exercise Sheet 5 Flowcharts and Pointers

### Exercise 1

For each of the following, write a single C statement that performs the indicated task.

- (a) Declare and initialise `next` to be a character variable with the value `'B'`.
- (b) Declare and initialise `current` to be a character variable with the value `'y'`.
- (c) Declare `ptr` to be a pointer to objects of type `char`.
- (d) Assign the address of `current` to the variable `ptr`.
- (e) Change the value of the object pointed to by `ptr` to `'0'`.
- (f) Assign the address of `next` to the variable `ptr`.
- (g) Change the value of the object pointed to by `ptr` to `'d'`.
- (h) Print the address stored in `ptr`.
- (i) What values are stored in `next` and `current`?

### Exercise 2

Given these declarations:

```
int x=0;
int y=0;
int* myPtr=NULL;
int* otherPtr=NULL;
```

Write down values of `myPtr`, `otherPtr`, `x`, and `y`, after each of the following lines of code:

- (a) `myPtr=&x;`
- (b) `otherPtr=&y;`
- (c) `*myPtr=4;`
- (d) `*otherPtr=*myPtr;`
- (e) `x=5;`
- (f) `otherPtr=myPtr;`
- (g) `*otherPtr=6;`

### Exercise 3

A family has a running account at a local milk bar. Every time a purchase is made, the amount of the purchase is subtracted from the running tally. When the next purchase costs more than the money remaining in the account, the sale cannot be made.

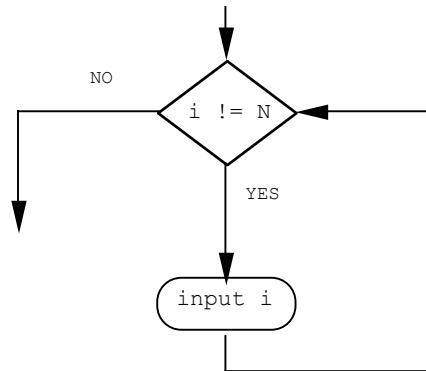
Write a C program that implements this policy. Your program should read the initial amount paid by the family. The account is updated by calling a function that receives two parameters: a pointer to the variable which contains the account, and the amount of the current purchase. It returns a Boolean value indicating whether the purchase can be made.

The program should print the amount remaining in the account after each sale, and it should stop with an appropriate message giving the amount remaining in the account when a requested sale cannot be made.

- (a) Complete the two gaps in the function prototype for the account update function:  
`___ update(___, float);`
- (b) give the C code for your program including the complete account update function.

**Exercise 4: Flow Diagrams**

Write the C code which corresponds to the algorithm represented by the following flow diagram (also called flow chart) assuming that *i* and *N* are integers.



**Exercise 5: Flow Diagrams**

Draw a flow diagram for the following C code fragment.

```

flag = 1;
for (i = 0; i < 10 && flag == 1; i++)
{
    if (i == 4)
    {
        flag = -1;
    }
    printf("%d ", i);
}
    
```

**Exercise 6: Test Data**

Consider the following C code.

```

int x, y;
scanf("%d %d", &x, &y);
while (x > y)
{
    printf("S1\n");
    x--;
    if (x == 4)
    {
        printf("S2\n");
    }
}
printf("S3\n");
    
```

- (a) Draw the flowchart for this fragment of C code.
- (b) Identify the instructions that are executed always.
- (c) Identify the choice points in the program and the flowchart.
- (d) Fill in the following table with valid test data (values for x and y) that tests all the paths in the program. Include the expected output for each test data.

Test Data	Expected Output