Ian Field SE1SA5

**Practical 3**

# Reviewing the Lecture Material

## Program 1

### Introduction

This program illustrates a use of printf(); to output 3 lines of text.

### Raw

#include <stdio.h>

/\* this program produces three

lines of output \*/

int main ( )

{

printf("This module is called: \n");

printf("\t Programming \n");

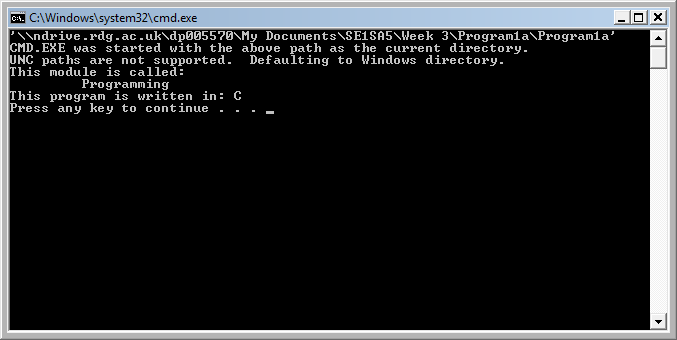
printf("This program is written in: ");

printf("C\n");

return 0; /\* returns successfully \*/

}

### Screenshot



### Experimental Code

#include <stdio.h>

/\* this program produces three

lines of output \*/

int main ( )

{

printf("This module is called: \n");

printf("\t Programming \n");

printf("This program is written in: ");

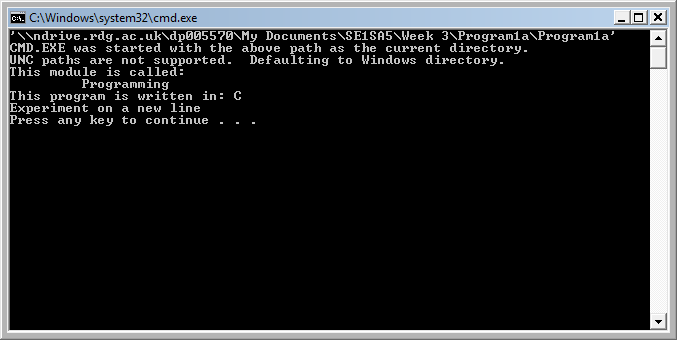
printf("C\n");

printf("Experiment on a new line\n");/\*Write a new line with text\*/

return 0; /\* returns successfully \*/

}

### Experimental Code Outcome



## Program 2

### Introduction

This program shows the printf(); function with parameters

### Raw

#include <stdio.h>

/\* illustrates printf \*/

int main ( )

{

int num =12;

char letter;

float fNum;

letter = 'A';

fNum = 12.34;

printf ("num is %d \n", num);

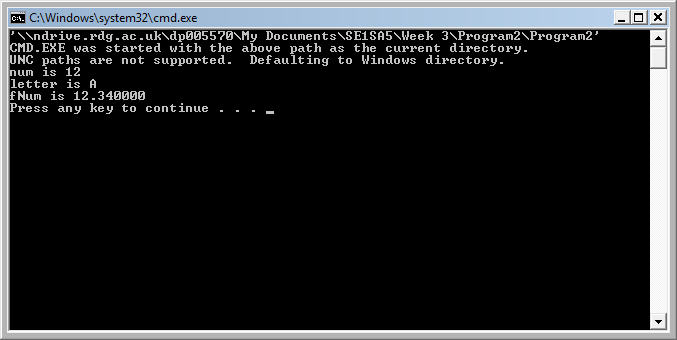
printf ("letter is %c \n", letter);

printf ("fNum is %f \n", fNum);

return 0;

}

### Screenshot



### Experimental Code

#include <stdio.h>

/\* illustrates printf \*/

int main ( )

{

int myNum =10;/\*Added\*/

int num =12;

char letter;

float fNum;

letter = 'A';

fNum = 12.34;

printf ("num is %d \n", num);

printf ("letter is %c \n", letter);

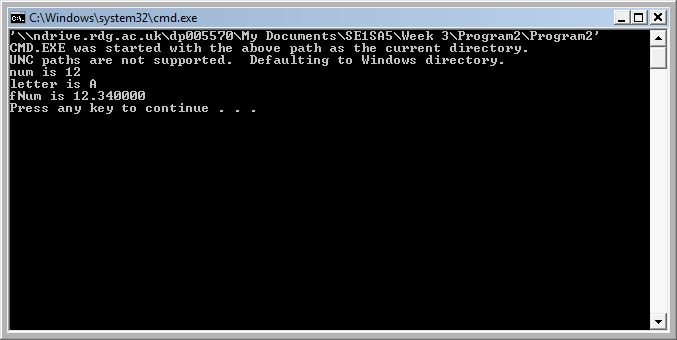
printf ("fNum is %f \n", fNum);

printf ("myNum is %d \n", myNum);/\*Output myNum\*/

return 0;

}

### Experimental Code Outcome



## Program 3

### Introduction

This program uses printf(); to display the maximum and minimum values that various integer types can take

### Raw

#include <stdio.h>

#include <limits.h>

int main ( )

{

short int small;

long int tall;

printf ("Max int:\t %d\n", INT\_MAX);

printf ("Max long int:\t %d\n", LONG\_MAX);

printf ("Max short int:\t %d\n", SHRT\_MAX);

printf ("Min int:\t %d\n", INT\_MIN);

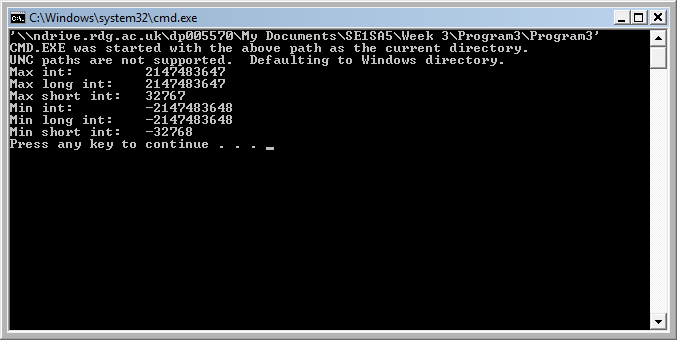
printf ("Min long int:\t %d\n", LONG\_MIN);

printf ("Min short int:\t %d\n", SHRT\_MIN);

return 0;

}

### Screenshot



### Experimental Code

#include <stdio.h>

#include <limits.h>

int main ( )

{

//short int small;

//long int tall;

printf ("Max int:\t %d\n", INT\_MAX);

printf ("Max long int:\t %d\n", LONG\_MAX);

printf ("Max short int:\t %d\n", SHRT\_MAX);

printf ("Min int:\t %d\n", INT\_MIN);

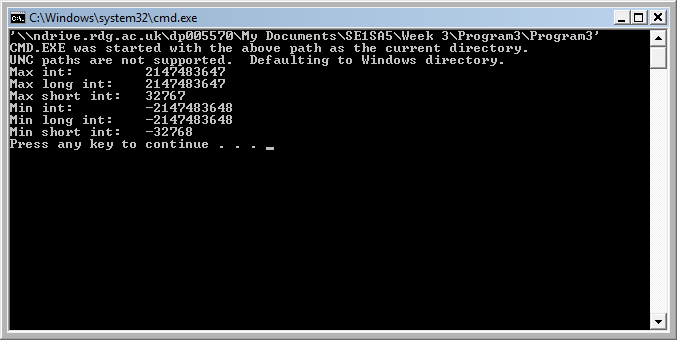
printf ("Min long int:\t %d\n", LONG\_MIN);

printf ("Min short int:\t %d\n", SHRT\_MIN);

return 0;

}

### Experimental Code Outcome



## Program 4

### Introduction

This program is used to demonstrate the use of external variables in a program

### Raw

#include <stdio.h>

int num; /\* define num \*/

int main ( )

{

extern int num; /\* explicitly declare num is needed\*/

num = 5; /\*use num\*/

/\* more code \*/

return 0;

}

/\* other function that can use num

also declares:

extern int num;

access the same num - values retained \*/

### Outcome

There is no screenshot as there is no visual output, however, the program executed as expected

### Experimental Code

#include <stdio.h>

int num; /\* define num \*/

int main ( )

{

extern int num; /\* explicitly declare num is needed\*/

num = 5; /\*use num\*/

/\* more code \*/

printf ("The value of num is now %d\n", num);

return 0;

}

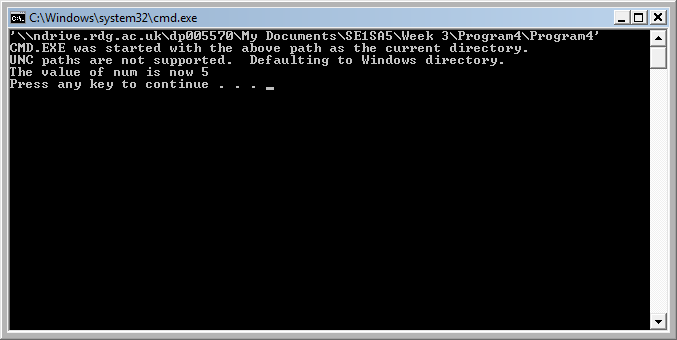
/\* other function that can use num

also declares:

extern int num;

access the same num - values retained \*/

### Experimental Code Outcome



## Program 5

### Introduction

This program demonstrates the ‘casting’ feature in the C language

### Raw

#include <stdio.h>

int main ( )

{

int i = 3;

char j = '1';

printf("int i is:%d \n" , i);

printf("char j is: %c \n", j);

i = (int) j; /\* cast j as an int \*/

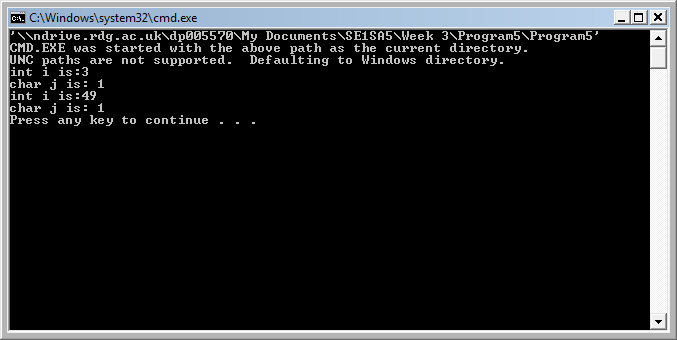
printf("int i is:%d \n" , i);

printf("char j is: %c \n", j);

return 0;

}

### Screenshot



### Experimental Code

#include <stdio.h>

int main ( )

{

int i = 3;

char j = '1';

char myChar = 'B';

int myInt = 2;

printf("int i is:%d \n" , i);

printf("char j is: %c \n", j);

i = (int) j; /\* cast j as an int \*/

myInt = (int) myChar; /\* cast myChar as int \*/

printf("int i is:%d \n" , i);

printf("char j is: %c \n", j);

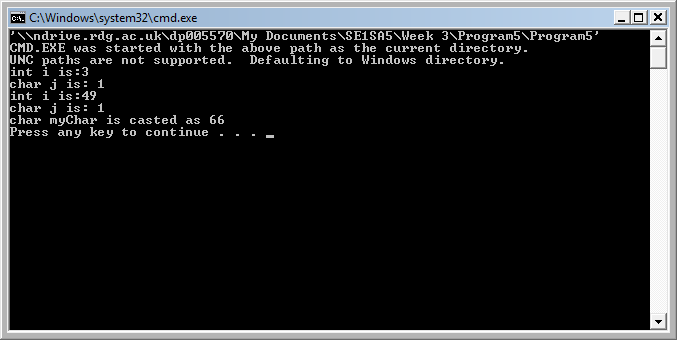
printf("char j is casted as %d \n", myInt);

return 0;

}

### Experimental Code Outcome

The character ‘B’ is casted to an integer



## Program 6

### Introduction

This program demonstrates how C uses arrays of characters to store strings

### Raw

#include <stdio.h>

int main ( )

{

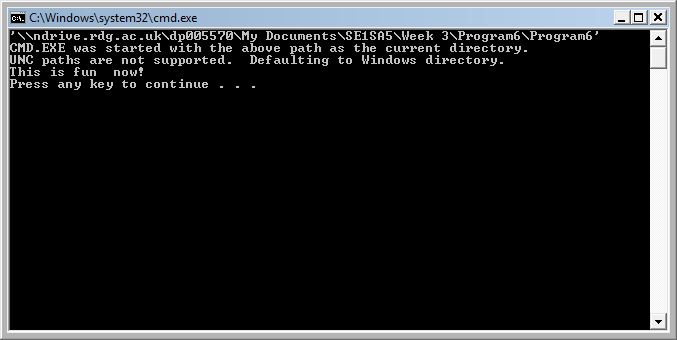
char Mood[5] ={'f','u','n', ' ','\0'};

printf ("This is %s now!\n", Mood );

return 0;

}

### Screenshot



### Experimental Code

#include <stdio.h>

int main ( )

{

char Mood[5] ={'f','u','n', ' ','\0'};

char myName[4] = {'I','a','n','\0'};

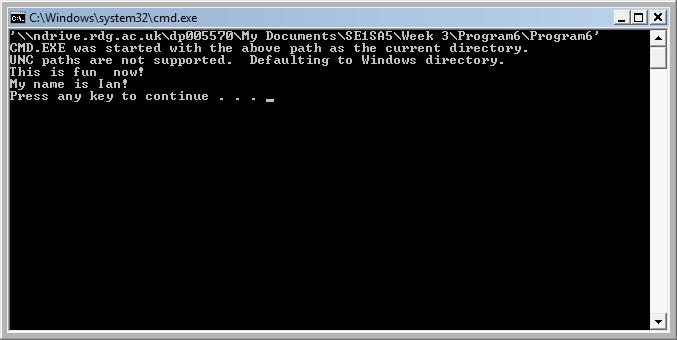
printf ("This is %s now!\n", Mood );

printf ("My name is %s!\n", myName);/\*Output myName\*/

return 0;

}

### Experimental Code Outcome



## Program 7

### Introduction

This program demonstrates the use of definitions in the pre-processor.

### Raw

#include <stdio.h>

#define FREEZE -32

#define BOIL 212

#define LINE "-o-o-o-o-o-o-o-o-\n"

int main ( )

{

printf("in farenheit freezing occurs at: %d \n",

FREEZE);

printf("%s", LINE);

printf("in farenheit boiling occurs at: %d \n",

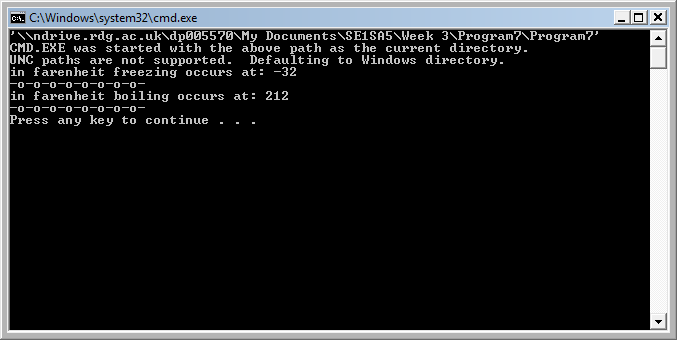
BOIL);

printf("%s", LINE);

return 0;

}

### Screenshot



### Experimental Code

#include <stdio.h>

#define FREEZE -32

#define BOIL 212

#define LINE "-o-o-o-o-o-o-o-o-\n"

#define NAME "Ian\n"

int main ( )

{

printf("in farenheit freezing occurs at: %d \n",

FREEZE);

printf("%s", LINE);

printf("in farenheit boiling occurs at: %d \n",

BOIL);

printf("%s", LINE);

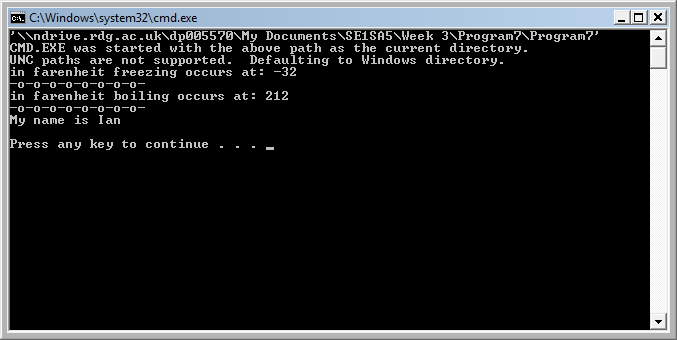
/\*Outputs NAME from the pre-processor\*/

printf("My name is %s\n", NAME);

return 0;

}

### Experimental Code Outcome



## Snippets

### Snippet 1

#### Introduction

This snippet shows how Variables can be used in a mathematical operation

#### Code

#include <stdio.h>

int main ()

{

int num1, num2;

int sum;

num1 = 2;

num2 = 5;

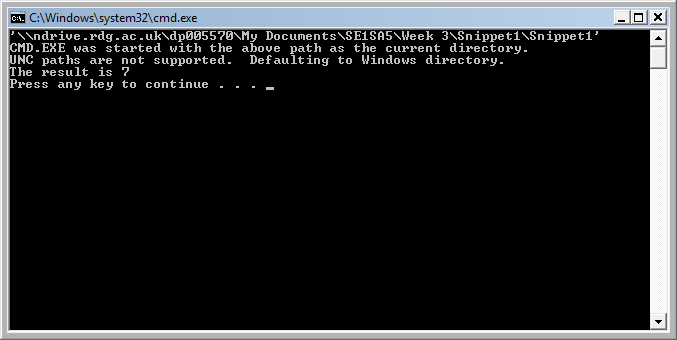
sum = num1 + num2;

printf("The result is %d\n", sum);

return 0;

}

#### Screenshot



### Snippet 2

#### Introduction

This snippet shows the use of scanf();

#### Code

#include <stdio.h>

int main ()

{

int num =12;

char letter;

float fNum;

printf ("please enter a character: ");

scanf ("%c", &letter);

printf ("please enter a float: ");

scanf ("%f", &fNum);

printf ("num is %d \n", num);

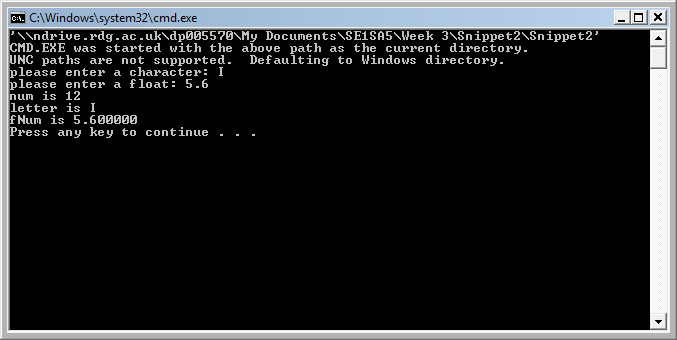
printf ("letter is %c \n", letter);

printf ("fNum is %f \n", fNum);

return 0;

}

#### Screenshot

******

### Snippet 3

#### Introduction

This is showing the use of value placement.

#### Code

#include <stdio.h>

int main ()

{

int num =12;

printf ("num using %%d is %d \n", num);

printf ("num using %%1d is %1d \n", num);

printf ("num using %%2d is %2d \n", num);

printf ("num using %%3d is %3d \n", num);

printf ("num using %%4d is %4d \n", num);

printf ("num using %%05d is %05d \n", num);

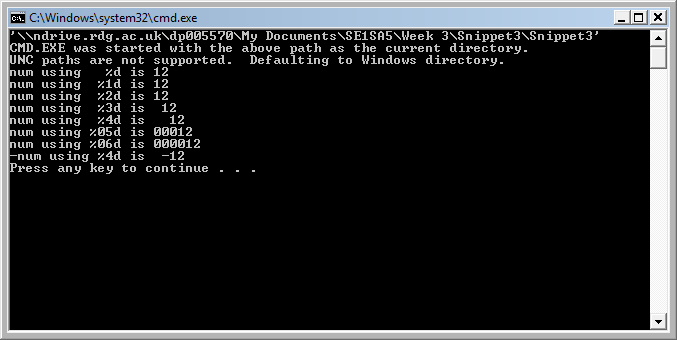
printf ("num using %%06d is %06d \n", num);

printf ("-num using %%4d is %4d \n", -num);

return 0;

#### }

#### Screenshot



### Snippet 4

#### Introduction

This demonstrates the use of the sizeof(); function for: Short int, float and double.

(slight alteration of slide snippet as ‘small’ and ‘tall’ are not included within the scope of the program)

#### Code

#include <stdio.h>

int main ()

{

printf("sizeof short int %d\n", sizeof (short int));

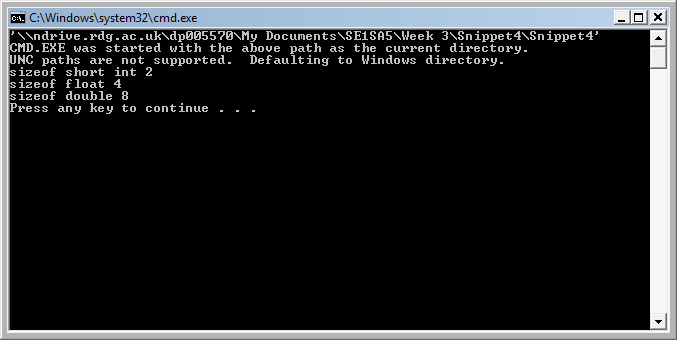
printf("sizeof float %d\n", sizeof (float));

printf("sizeof double %d\n", sizeof (double));

return 0;

#### }

### Screenshot



### Snippet 5

#### Introduction

This program demonstrates the use of the enum datatype creation.

#### Code

#include <stdio.h>

int main ()

{

enum boolean {FALSE, TRUE};

printf ("value of False is: %d\n", FALSE);

printf ("value of True is: %d\n", TRUE);

return 0;

}

### Screenshot

### 

# Banner Print

## Analysis

It is required that the string ‘SE1SA5’ is to be outputted as a banner in the style that the character is made up of smaller versions of itself.

## Design

To do this the characters need to be outputted as a string in the console each line making up a row of the letter required to be created on-screen.

## Pseudo Code

Print(“SSSSS”);

Print(NEW LINE);

Print(“S “);

Print(NEW LINE);

Print(“ SSSSS”);

Print(NEW LINE);

Print(“ S”);

Print(NEW LINE);

Print(“ SSSSS”);

Print(NEW LINE);

Print(NEW LINE);

Print(“EEEEE”);

Print(NEW LINE);

Print(“E”);

Print(NEW LINE);

Print(“EEE”);

Print(NEW LINE);

Print(“E”);

Print(NEW LINE);

Print(“EEEEE”);

Print(NEW LINE);

Print(NEW LINE);

Print(“ 1”);

Print(NEW LINE);

Print(“1”);

Print(NEW LINE);

Print(“1”);

Print(NEW LINE);

Print(“1”);

Print(NEW LINE);

Print(NEW LINE);

Print(“AAAAA”);

Print(NEW LINE);

Print(“A A”);

Print(NEW LINE);

Print(“AAAAA”);

Print(NEW LINE);

Print(“A A”);

Print(NEW LINE);

Print(“A A”);

Print(NEW LINE);

Print(NEW LINE);

Print(“55555”);

Print(NEW LINE);

Print(“5 “);

Print(NEW LINE);

Print(“ 55555”);

Print(NEW LINE);

Print(“ 5”);

Print(NEW LINE);

Print(“ 55555”);

## Implementation

### Code

#include <stdio.h>

int main ()

{

printf("SSSSS\n");

printf("S \n");

printf("SSSSS\n");

printf(" S\n");

printf("SSSSS\n\n");

printf("EEEEE\n");

printf("E\n");

printf("EEE\n");

printf("E\n");

printf("EEEEE\n\n");

printf(" 11\n");

printf(" 11\n");

printf(" 11\n");

printf(" 11\n\n");

printf("AAAAA\n");

printf("A A\n");

printf("AAAAA\n");

printf("A A\n");

printf("A A\n\n");

printf("55555\n");

printf("5 \n");

printf("55555\n");

printf(" 5\n");

printf("55555\n\n");

return 0;

}

### Screenshot

