Errors

The following program has a critical problem (will not compile), can you identify what it is?

```
#include<stdio.h>

int main
{
 int a=1;
 b=1.3;
 a=a+b;

 printf("Variable a is %d and variable b is %d", a, b);

 return 0;
}
```

- variable 'b' is never declared
- You can't add a decimal number ('b') to an int variable ('a')
- Variable b can't be output through printf using the %d placeholder as it is not an integer.
- Declaring variable a and assigning its value at the same time (a=1) is not allowed.
- This program should run just fine, there are no critical issue with it.

Operators

The ++ operator is unique of C languages.

What does the expression `x++;` mean?

- x=x+1;
- x=x+x;
- x=x+2;

Expand the bit size of variable x, so it can contain larger numbers

Program syntax

There is a problem with the following program

I would expect b to take the value 1 but I get a value I cannot recognise as meaningful as the output of my printf statement.

What have I done wrong?

#include<stdio.h>

int main()
{
int a;
float b;

a=1;
b=b+a;
printf("Variable b is %f", b);
}

- Adding a float and an integer will cause you trouble
- The wrong type identifier is used in printf
- Variable b is not initialised
- return 0; is missing from the program

Program syntax 2

Is something wrong with the following simple program?

```
#include<stdio.h>
int main()
{
 int a,b;

 printf("Give me two numeric values!");
 scanf("%d %d", a,b);
 printf("\nThe sum of your two values is %d, now go eat some pizza!", a+b);
 return 0;
}
```

There is a problem with variable declarations/initialisation

There is an issue with one of the printf statements.

There is an issue with the scanf statement;

No, there is nothing wrong.

There's no pizza left

Structure of a C program

Which of the following is required in every single C program?

an #include directive

a main() function

variable declarations

Comments

Boolean operator ==

If variable a=1 and variable b=2

computing variable x=(a==b)

results in:

x=1

x=0

x=2

Results in an error

Boolean operators?

If variable a=1 and variable b=2

computing variable x=(a=b)

results in:

x=1

x=0

x=2

Results in an error

Operators

Which of the following is not a valid C operator?

++

!=

>>

<>

&&

+=

Program syntax 3

Is something wrong with the following simple program?

```
#include<stdio.h>
int main()
{
 int a,b;
 printf("Give me two numeric values!");
 scanf("%d %d", a,b);
 printf("\nThe product of your two values is %d", a*b);
 return 0;
}
```

- There is a problem with variable declarations/initialisation
- There is an issue with one of the printf statements.
- There is an issue with the scanf statement;
- No, there is nothing wrong.

Special C arithmetic syntax

What will the values of the variables be at the end of this snippet of code?

```
int a,b,c=0;

a=1;
b=2;
b+=a;
b*=2;
c=(b>a);
c--;
```

- a=1 b=6 c=0

a=3 b=4 c=0

a=3 b=4 c=-1

a=1 b=2 c=5

a=1 b=1 c=0

Special C arithmetic syntax

What will the values of the variables be at the end of this snippet of code?

```
int a,b,c=0;

a=1;
b=2;
b+=a;
b*=2;
c=(b>a);
c--;
```

a=1 b=6 c=0

a=3 b=4 c=0

a=3 b=4 c=-1

a=1 b=2 c=5

a=1 b=1 c=0

Submit

Control instruction

How many times is "PRINT" printed to screen by the following program?

```
#include<stdio.h>
int main()
{
    int i;
    for(i=-1; i<=10; i++)
    {
        if(i < 5)
            continue;
        else
            break;
        printf("PRINT");
    }
    return 0;
}
```

- Infinite times
- 0 times
- 10 times
- 11 times
- 5 times

For loop

What will the output of the following code be?

```
#include <stdio.h>
void main()
{
int k;
for (k = -3; k < -5; k++)
printf("Hello");
}
```

- Hello

A very large number of Hello (requires variable k to be incremented to rollover and beyond before the loop will stop)

Nothing

Error

For loop

What will the output of the following code be?

```
#include <stdio.h>
void main()
{
  int k;
  for (k = -3; k < -5; k++)
     printf("Hello");
}
```

Hello

A very large number of Hello (requires variable k to be incremented to rollover and beyond before the loop will stop)

Nothing

Error

if statement

What is the output of the following C program?

```
#include <stdio.h>
int main()
{
 int x = 1;
 if (x > 0)
   printf("inside if\n");
 else if (x > 0)
   printf("inside elseif\n");
}
```

- inside if
- inside elseif
- inside if
inside elseif
- Error

logical operator

Which of the following is not a logical operator?

- &&
- ==
- &
- !=

While loop

How many times will the following print out "PRINT"?

```
#include<stdio.h>
int main()
{
    char x='A';
    while(x)
    {
        printf("PRINT");
    }
    return 0;
}
```

- 0 times

infinite times

error since x is a character

65 times

While loops

What is the output of the following C code?

```
#include <stdio.h>
int main()
{
   int i = 0;
   while (i < 10)
   {
      i++;
      printf("hi\n");
      while (i < 8) {
         i++;
         printf("hello\n");
      }
   }
}
```

Hi is printed 8 times, hello 7 times and then hi 2 times

Hi is printed 10 times, hello 7 times

Hi is printed once, hello 7 times

Hi is printed once, hello 7 times and then hi 2 times

Submit

function returns again

What will the output of this program be?

```
#include <stdio.h>

 void m()
 {
 printf("hello");
 return 5;
 }

 int main()
 {
 int k = m();
 printf("%d", k);
 }
```

- 5
- hello
- hello 5
- Error

Recursion

How many times will the following program print "ENGN3213"?

```
#include<stdio.h>

int main()
{
printf("ENGN3213");
main();
return 0;
}
```

- Infinite times
- 32767 times
- 65535 times

Many times, until the stack overflows.

return type

What will the output of this program be?

```
#include <stdio.h>
#include <math.h>

int func(void)
 {
 return sqrt(2.25);
 /*sqrt is a floating point function returning the square root of a number*/
 }

int main()
 {
 float x = 0;
 x = func();
 printf("%f", x);
 return 0;
 }
```

0.000000

1.000000

1.500000

Error

Scope of variables

What will the output of the following code be?

```
#include <stdio.h>

int x()
{
int a=2;
printf("Hi");
return 5;
}

int main()
{
int a=0;
x();
```

```
printf(" %d", a);
}
```

- Hi 0
- Hi 2
- Hi 5
- Error

Scope of variables

What will the output of the following code be?

```
#include <stdio.h>

int x()
{
    int a=2;
    printf("Hi");
    return 5;
}

int main()
{
    int a=0;
    x();
    printf(" %d", a);
}
```

- Hi 0
- Hi 2
- Hi 5
- Error

Recursion again

What number will this program print out?

```
#include <stdio.h>

 int fcn(int a)
 {
 if (a==1)
 return a;
 else
 return a*fcn(a-1);
 }

 int main()
 {
 int a=0;
 a=fcn(5);
 printf(" %d", a);
 }
```

Answer

Submit

array and pointer arithmetic

What is the output of the code below

```
#include <stdio.h>
int main()
{
int ary[4] = {1, 2, 3, 4};
int *p = ary + 3;
printf("%d\n", p[-2]);
}
```

- 1
- 2
- Error
- Undefined value

pointer and array again

What will the output of the following code be?

```
#include <stdio.h>
int main()
{
int ary[4] = {1, 2, 3, 4};
int* p;
p=ary+1;
printf("%d %d", ary[3], p[3]);
}
```

- 4 5
- 4 followed by an undefined number
- error

two different addresses (cast as integer numbers due to the use of %d)

pointer function

What will the output of the following program be?

```
#include <stdio.h>
 int main()
 {
 int *ptr, a = 10;
 ptr = &a;
 *ptr += 1;
 printf("%d,%d/n", *ptr, a);
 }
```

- 10, 10
- 10, 11
- 11, 10
- 11, 11

pointer meaning

We have learned that a pointer stores a memory address.

In light of this, for a 32-bit machine (many general-purpose computers are 32-bit machines), you can expect the size of a pointer to be... ?

- 2 byte
- 4 byte
- 1 byte
- 8 byte

swap example

What will be the output of the following C program?

```
#include <stdio.h>
void m(int p, int q)
{
int temp = p;
p = q;
q = temp;
}

int main()
{
int a = 6, b = 5;
m(a, b);
printf("%d %d\n", a, b);
}
```

- 6 5
- 6 6
- 5 6
- 5 5
- Error

Value vs address

What is the output of this C code?

```
#include <stdio.h>
int x = 0;
int main()
{
int *ptr = &x;
printf("%p\n", ptr);
x++;
printf("%p\n ", ptr);
}
```

- Two identical addresses
- Two different addresses
- Error
- 0 1

value vs address #2

What is the output of this C code?

```
#include <stdio.h>
int main()
{
int x = 0;
int *ptr = &x;
printf("%p ", ptr);
ptr++;
printf("%p\n ", ptr);
}
```

- 0 1
- 0xbfd605e8 0xbfd605ec
- 0xbfd605e8 0xbfd605e8
- 0xbfd605e8 0cbfd60530
- Error

value vs address #3

What will be the output of this C program?

```
#include <stdio.h>
int main()
{
int x = 0;
int *ptr = &x;
printf("%d ", *ptr);
ptr++;
printf("%d\n ", *ptr);
}
```

- 0 1
- 0 followed by an undefined number
- Two different addresses
- Two equal numbers

Submit

functions and struct

What will the output of the following be?

```
#include <stdio.h>
struct temp
{
  int a;
} ;

void func(struct temp s)
{
  s.a = 10;
  printf("%d\t", s.a);
}

int main()
{
  struct temp s;
  func(s);
  printf("%d\t", s.a);
}
```

- 10 10
- 10 followed by a junk value
- a junk value followed by 10
- 10 0
- 0 10

functions and struct 2

What will the output of the following be?

```
#include <stdio.h>
struct temp
{
  int a;
} ;
```

```
void func(struct temp *s)
{
  *s.a = 10;
  printf("%d\t", *s.a);
}

int main()
{
  struct temp s;
  func(&s);
  printf("%d\t", s.a);
}
```

- 10 10
- 10 followed by a junk value
- Error
- 10 0
- 0 10

size of struct

Considering the size of an int as 2 bytes, and char representing the standard 256 ASCII character encoding, the memory size of the following is:

```
struct student
{
  int no;
  char name[20];
};
```

- 22 bytes
- greater than 22 bytes
- smaller than 22 bytes

it depends on the values stored in the structure members

structure access

Given the structure

```
struct
{
  int x;
  int y;
} abc;
```

which of the following ways to assign x is/are INVALID?

1. abc -> x = 1;
2. abc[0].x = 1;
3. abc.x = 1;
4. (abc) ->x = 1;

Option 1,2 and 4

Option 2 and 3

Option 1 and 3

Option 1,3 and 4

structure access

Given the structure

```
struct
{
  int x;
  int y;
} abc;
```

which of the following ways to assign x is/are INVALID?

1. abc -> x = 1;
2. abc[0].x = 1;
3. abc.x = 1;
4. (abc) ->x = 1;

- Option 1,2 and 4
- Option 2 and 3
- Option 1 and 3
- Option 1,3 and 4

syntax 1

What will the output of the following C program be?

```
#include <stdio.h>
struct student
{
  int no;
  char name[20];
};

int main()
{
  struct student;
  no = 8;
  printf("%d", no);
}
```

- Error
- 8
- 0
- Junk value
- Nothing

syntax 2

What will the output of the following program be?

```
#include <stdio.h>
struct student
{
  int no;
  char name[20];
}

int main()
{
  struct student s;
  s.no = 8;
  printf("hello", s.no);
}
```

- Error
- 8
- hello
- None of the options mentioned

Submit