You may use one page of hand written notes (both sides) and a dictionary.
No i-phones, calculators nor any other type of non-organic computer.

**1) If Logic:** This C program compiles and runs. What is its output?

```c
#include <stdio.h>
void main(void)
{
  int a = 14;
  if (a % 7 == 0)
  { printf("A\n");
    a+=2;
  }
  else
  { printf("B\n");
    a+=4;
  }
  printf("C\n");
  if (a % 7 == 0)
  { printf("D\n");
    a+=3;
  }
  else
  { printf("E\n");
    if (a > 5)
    { printf("F\n");
      a +=4;
    }
    else if (a > 10)
    { printf("G\n");
      a -=4;
    }
  }
  printf("%d\n",a);
}
```

**2) Variable Scope:** This C program compiles and runs. What is its output?

```c
#include <stdio.h>

int b=5;

int foo(int n)
{
  int a=3;
  a++;
  b++;
  n = n+a+b;
  printf("foo: n=%d, a=%d, b=%d \n", n, a, b);
  return n;
}

void main(void)
{
  int a, n;
  n = 5;
  a = foo(n);
  printf("main: n=%d, a=%d, b=%d\n", n, a, b);

  a = foo(n);
  printf("main: n=%d, a=%d, b=%d\n", n, a, b);
}
```

**3) Binary Search:** This C program compiles and runs. What is its output?

```c
#include <stdio.h>

#define ENDCODE -1
int binarySearch(int x, int v[])
{
  int mid;
  int low = 0;
  int high = 0;
  while (v[high] != ENDCODE) high++;
  high--;
  while (low <=high)
  {
    mid = (low+high)/2;
    printf("[%d %d %d] ", low, mid, high);

    if (x < v[mid]) high = mid-1;
    else if (x > v[mid]) low = mid+1;
    else return mid;
  }
  return -1;
}

void main(void)
{
  int nums[]={12, 13, 15, 17, 21, 23, 27, 39, 43, 51, -1};
  printf("idx = %d\n", binarySearch(21, nums));
  printf("idx = %d\n", binarySearch(30, nums));
}
```

**4) Bit Operators**: This *C* program compiles and runs. What is its output?

```c
#include <stdio.h>
void main(void)
{ unsigned char x = 60;

  unsigned char a = x << 4;
  unsigned char b = x >> 4;
  unsigned char c = x & 15;
  unsigned char d = x & 240;
  unsigned char e = x | 15;
  unsigned char f = x ^ 15;

  printf("a=%d, b=%d, c=%d, d=%d, e=%d, f=%d\n",
          a, b, c, d, e, f);
}
```

**6) Squeeze: removing a character from a string in place.** This C program compiles and runs. What is its output?

```
#include <stdio.h>

void main(void)
{
  char s[]="XbXXytXXXe";
  char del ='X';
  int srcIdx=0, snkIdx=0;
  while (s[srcIdx])
  { if (s[srcIdx] != del)
    { s[snkIdx] = s[srcIdx];
      snkIdx++;
    }
    else
    { printf("[%d,%d] %s\n", srcIdx, snkIdx, s);
    }
    srcIdx++;
  }
  s[snkIdx]='\0';
  printf("==>%s\n",s);
}
```

**7)** This C program compiles and runs. What is its output?

```c
#include <stdio.h>

void main(void)
{

  char bits[40];
  bits[39] = '\0';
  unsigned int n=400;
  unsigned int p;
  int i;
  int k=38;
  for (i=0; i<32; i++)
  {
    unsigned int p=1<<i;
    if (n & p) bits[k] = '1';
    else bits[k] = '0';
    if ((i+1) % 4 == 0) bits[--k] = '-';
    k--;
  }

  printf("%s\n", bits);
}
```