COMP 2401
Test #1

[out of 50 marks]

1. [6 marks]

   a. [2 marks]
   Answer: `0x5f = 5*16^1 + 15 = 95 + 77 = 172`
   Marking:
   -- 1 mark for showing expansion using powers of 16
   -- 1 mark for correct answer

   b. [2 marks]
   Answer: 172 = 2^7 + 2^5 + 2^3 + 2^2 = **`1010 1100`**
   Marking:
   -- 1 mark for showing expansion using powers of 2
   -- 1 mark for correct answer

   c. [2 marks]
   Answer #1: **`1010 1100`**
   -- interpreted as memory representation, answer is identical to (b)
   Answer #2: **`0101 0100`**
   -- interpreted as what is printed out, answer is as follows
   -- apply two's complement to **`1010 1100`**
   ---- invert: **`0101 0011`**
   ---- add 1: **`0101 0100`**
   Marking:
   -- 1 mark for applying two's complement
   -- 1 mark for correct answer

2. [44 marks]

   a. [4 marks]

```
typedef struct {          // 1 mark for typedef struct and HeroType
  char name[MAX_STR];        // 1 mark for name
  char avatar;            // 1 mark for avatar
  PositionType pos;       // 1 mark for pos
} HeroType;
```

b. [6 marks]

i. [3 marks]
Answer: Yes, the program would compile because there is nothing wrong with its syntax
Marking:
-- 1 mark for correct answer
-- 2 marks for explanation

ii. [3 marks]
Answer: No, the program would not execute correctly because the hero structures have not been initialized. This will result in a segmentation fault at some point in the program.
Marking:
-- 1 mark for correct answer
-- 2 marks for explanation

c. [6 marks]

i. [2 marks]
Answer: by reference
Marking: all or nothing

ii. [2 marks]
Answer: by reference
Marking: all or nothing

iii. [2 marks]
Answer: by value
Marking: all or nothing

d. [6 marks]

i. [3 marks]
Answer: input/output
Marking: all or nothing

ii. [3 marks]
Answer: input/output or output
Marking: all or nothing

e. [4 marks]:

```
// 2 marks for prototype:
// -- 1 mark for taking HeroType* as parameter
// -- 1 mark for returning int

int  heroWins(HeroType *hero)
{
// 1 mark for comparing hero's y coordinate to zero
// 1 mark for returning this result (or C_TRUE/C_FALSE)
  return (hero->pos.y == 0);
}
```

f. [18 marks]

```
// 2 marks for parameters
// -- 1 mark for HeroType* for hero
// -- 1 mark for int* for new x-coordinate
void computeNewX(HeroType *hero, int *x)
{
// 1 mark for calling random
  int rnum = random(3);

// 3 marks for switch or if-else, 1 mark for each branch
  switch(rnum) {
    case 0:
// 3 marks for new x-coordinate in first branch
// -- 1 mark for using hero's current x-coordinate
// -- 1 mark for decrementing it by 1
// -- 1 mark for setting *x (zero if not dereferenced)
      *x = hero->pos.x - 1;
      break;
    case 1:
// 3 marks for new x-coordinate in second branch
// -- 1 mark for using hero's current x-coordinate
// -- 1 mark for incrementing it by 1
// -- 1 mark for setting *x (zero if not dereferenced)
      *x = hero->pos.x + 1;
      break;
    case 2:
// 2 marks for new x-coordinate in third branch
// -- 1 mark for using hero's current x-coordinate
// -- 1 mark for setting *x (zero if not dereferenced)
      *x = hero->pos.x;
  }

// 2 marks for checking that new x-coordinate is not negative
  if (*x < 0)
    *x = 0;
// 2 marks for checking that new x-coordinate is not greater than max
  if (*x > MAX_X - 1)
    *x = MAX_X- 1;
}
```